US006857116B1

(12) **United States Patent**
Dahl et al.

(10) **Patent No.: US 6,857,116 B1**
(45) **Date of Patent: Feb. 15, 2005**

(54) **OPTIMIZATION OF ABUTTED-PIN HIERARCHICAL PHYSICAL DESIGN**

(75) Inventors: **Peter Dahl**, Cupertino, CA (US); **Byron Dickinson**, San Jose, CA (US); **Margie Levine**, Menlo Park, CA (US); **Paul Rodman**, Palo Alto, CA (US)

(73) Assignee: **Reshape, Inc.**, Mountain View, CA (US)

(21) Appl. No.: **09/714,722**

(22) Filed: **Nov. 15, 2000**

(51) **Int. Cl.⁷** **G06F 17/50**
(52) **U.S. Cl.** **716/12**; 716/13; 716/14
(58) **Field of Search** 716/2, 4, 7, 8, 716/9, 11, 12, 13, 14

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,890,238 A		12/1989	Klein et al.	364/491
5,187,671 A		2/1993	Cobb	364/490
5,309,370 A		5/1994	Wong	364/490
5,533,148 A		7/1996	Sayah et al.	382/240
5,544,088 A		8/1996	Aubertine et al.	364/489
5,576,969 A		11/1996	Aoki et al.	364/491
5,757,658 A		5/1998	Rodman et al.	364/491
6,243,854 B1		6/2001	Lavin et al.	716/19
6,425,113 B1	*	7/2002	Anderson et al.	716/5
6,618,849 B2		9/2003	Teig et al.	716/12

FOREIGN PATENT DOCUMENTS

WO	WO00/67163	11/2000	G06F/17/50

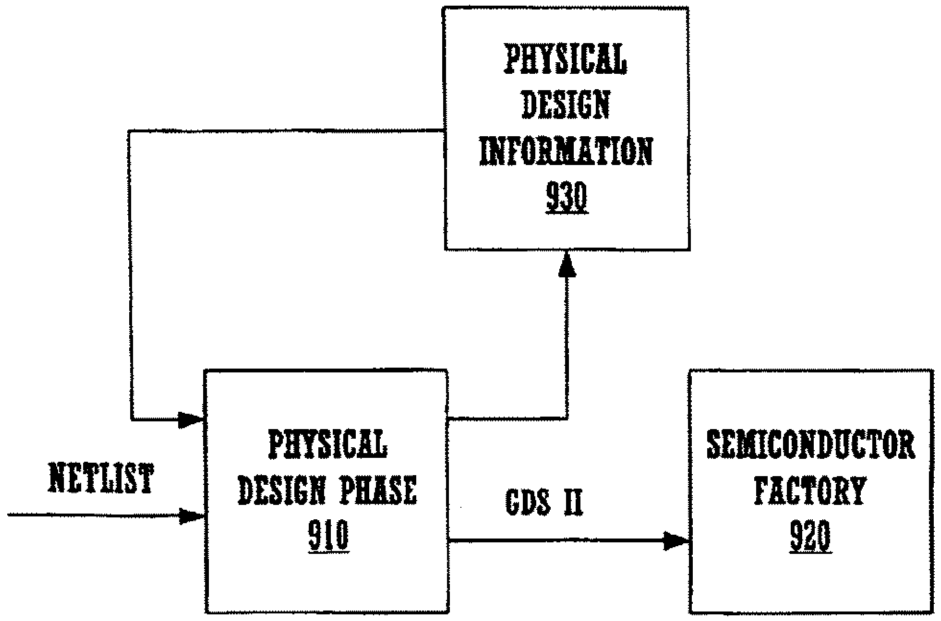
* cited by examiner

Primary Examiner—Thien F Tran
(74) *Attorney, Agent, or Firm*—Wagner, Murabito, & Hao LLP

(57) **ABSTRACT**

An abutted-pin hierarchical physical design process is described. The abutted-pin hierarchical physical design provides solutions to the problems of the traditional hierarchical physical design and provides additional advantages and benefits. In particular, the abutted-pin hierarchical physical design does not have channels. Moreover, in the abutted-pin hierarchical physical design, components of the top-level are merged into the block-level so that the top-level netlist is reduced significantly.

52 Claims, 31 Drawing Sheets


PHYSICAL DESIGN INFORMATION 930
NETLIST
PHYSICAL DESIGN PHASE 910
GDS II
SEMICONDUCTOR FACTORY 920

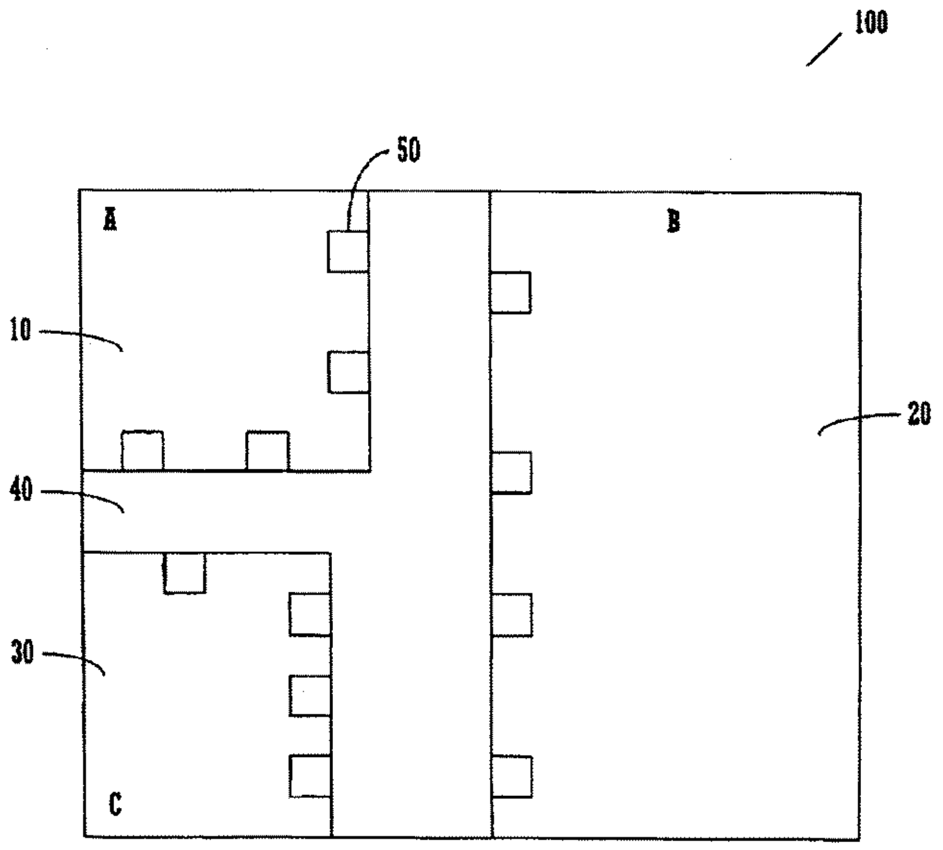


FIGURE 1
(Prior Art)

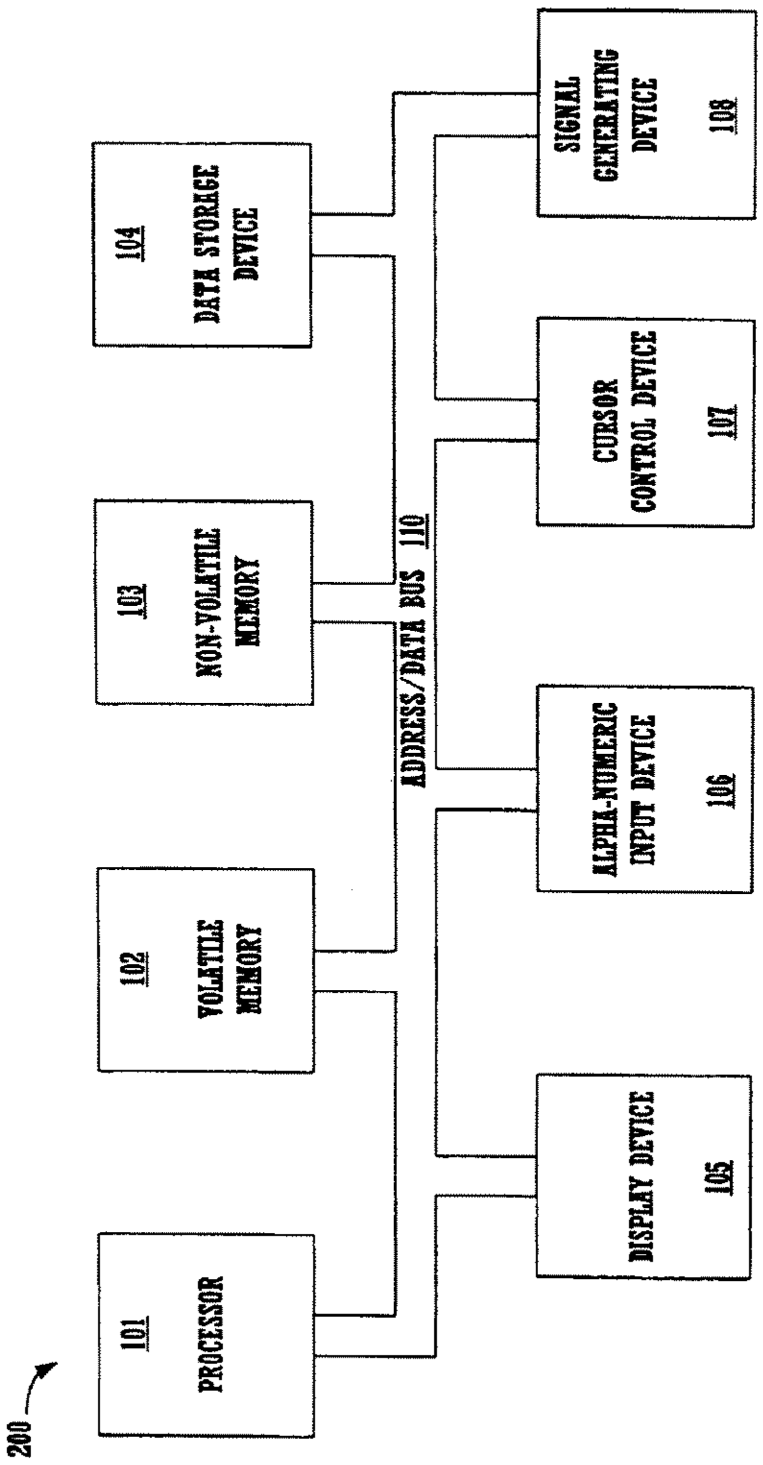

200
101
PROCESSOR
102
VOLATILE MEMORY
103
NON-VOLATILE MEMORY
104
DATA STORAGE DEVICE
ADDRESS/DATA BUS 110
DISPLAY DEVICE
105
ALPHA-NUMERIC INPUT DEVICE
106
CURSOR CONTROL DEVICE
107
SIGNAL GENERATING DEVICE
108

FIGURE 2

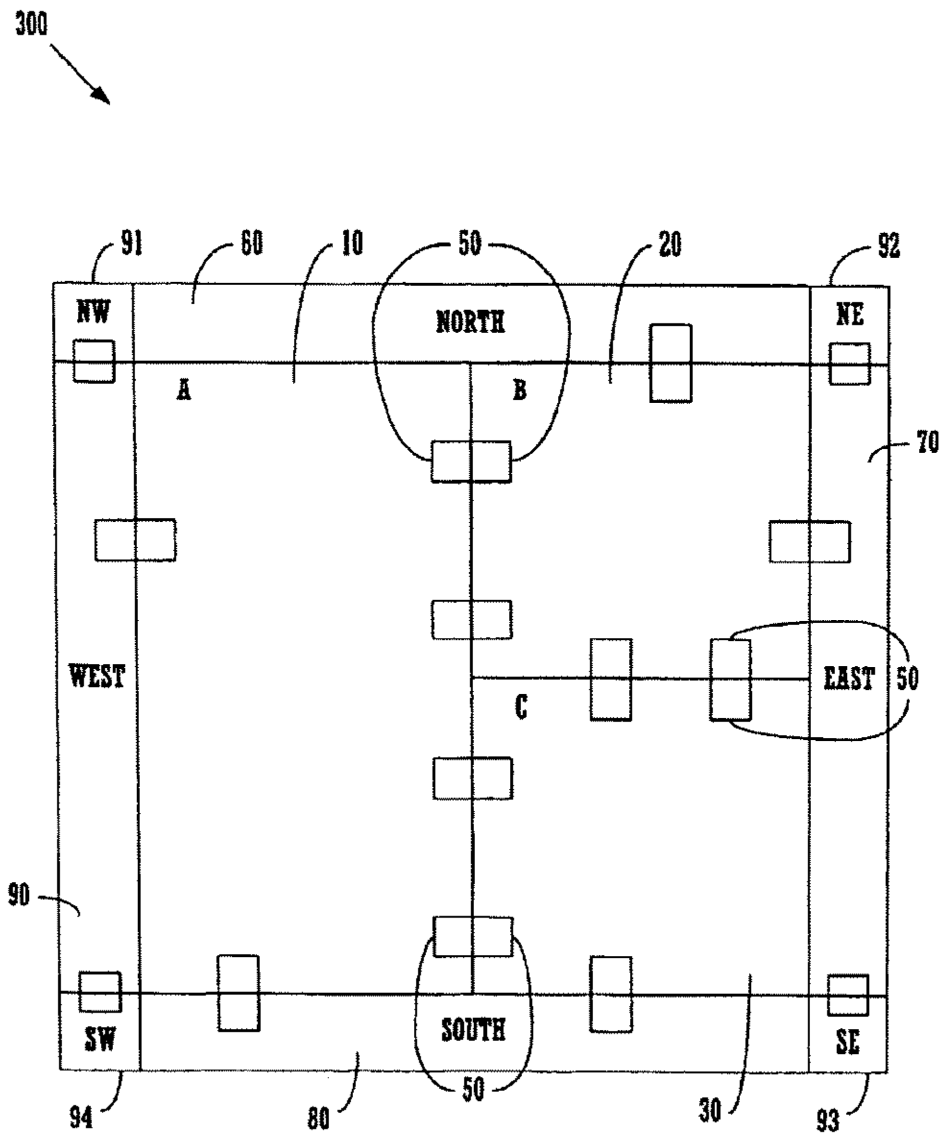

300
91
60
10
50
20
92
NW
NORTH
NE
A
B
70
WEST
C
EAST
50
90
SW
SOUTH
SE
50
94
80
30
93

FIGURE 3

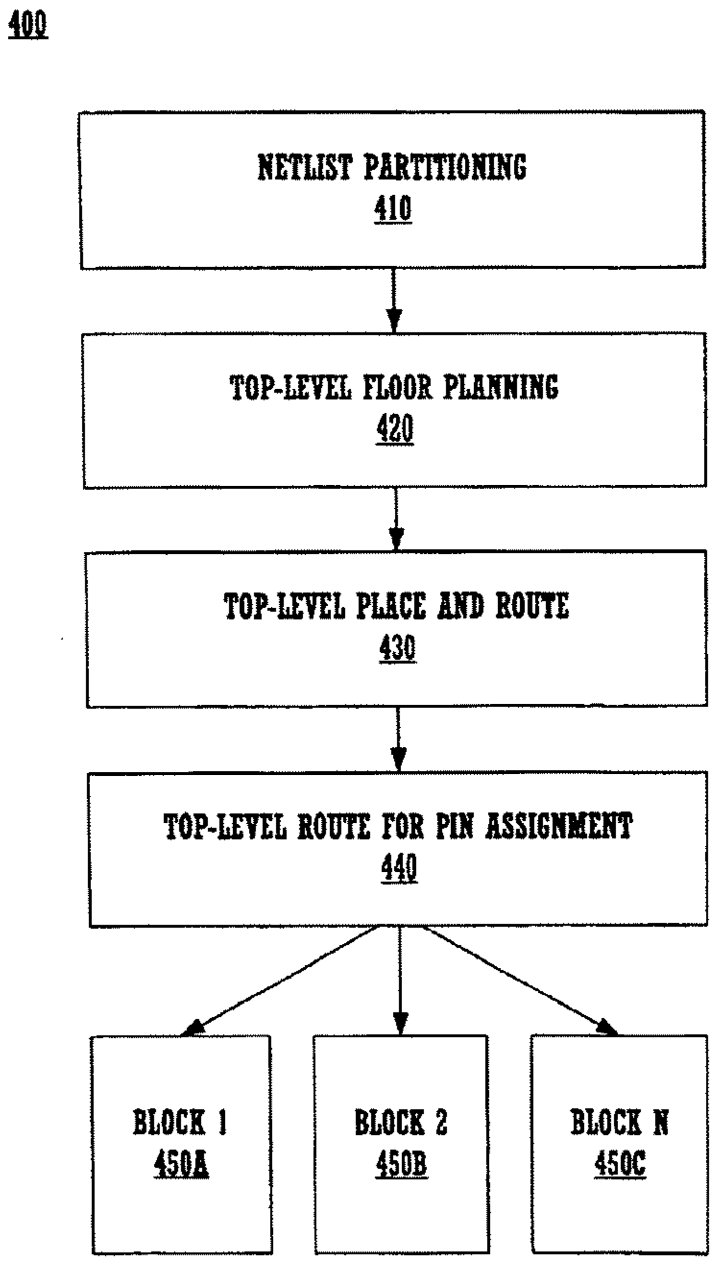

400
NETLIST PARTITIONING
410
TOP-LEVEL FLOOR PLANNING
420
TOP-LEVEL PLACE AND ROUTE
430
TOP-LEVEL ROUTE FOR PIN ASSIGNMENT
440
BLOCK 1
450A
BLOCK 2
450B
BLOCK N
450C

FIGURE 4

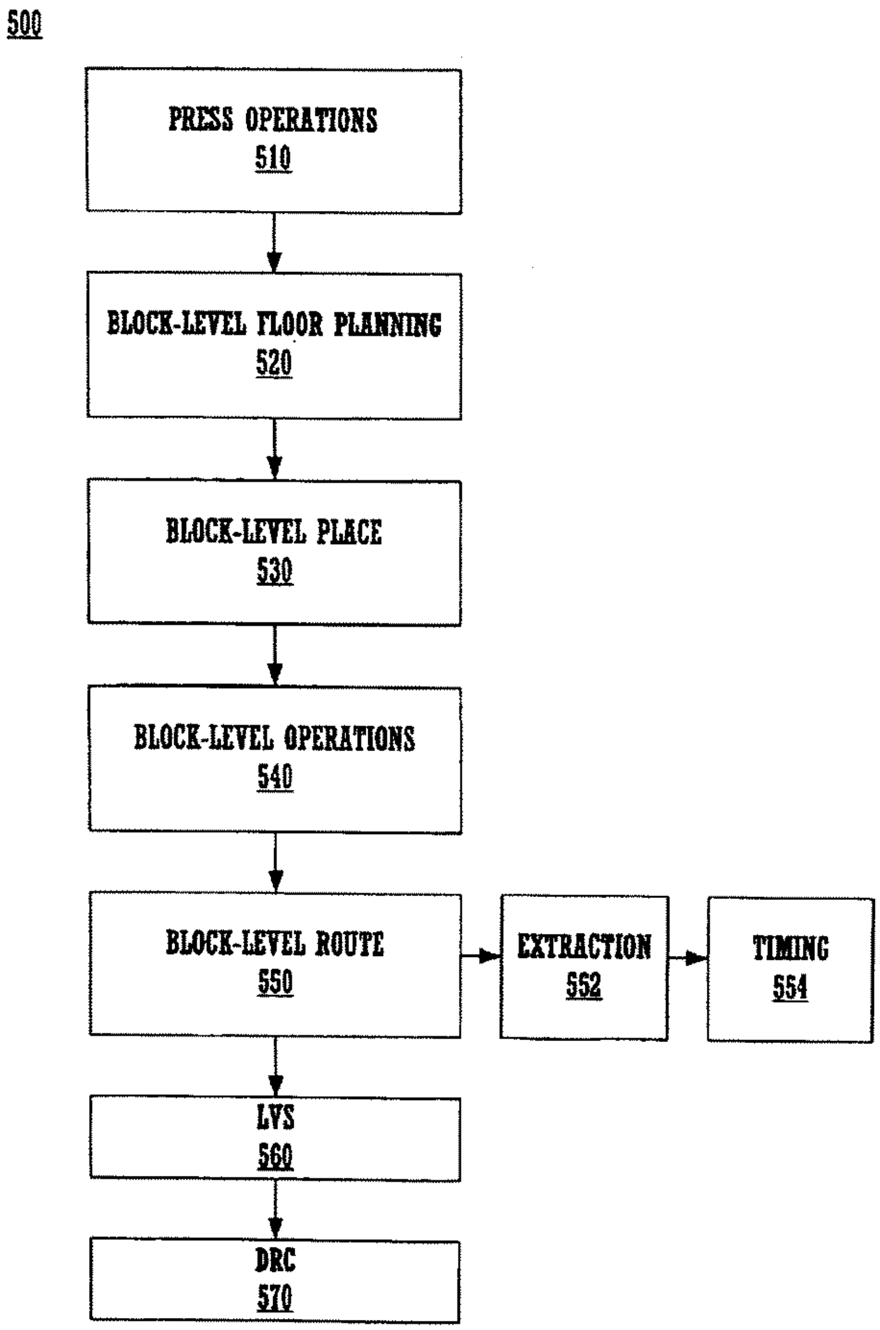

500
PRESS OPERATIONS 510
BLOCK-LEVEL FLOOR PLANNING 520
BLOCK-LEVEL PLACE 530
BLOCK-LEVEL OPERATIONS 540
BLOCK-LEVEL ROUTE 550
EXTRACTION 552
TIMING 554
LVS 560
DRC 570

FIGURE 5

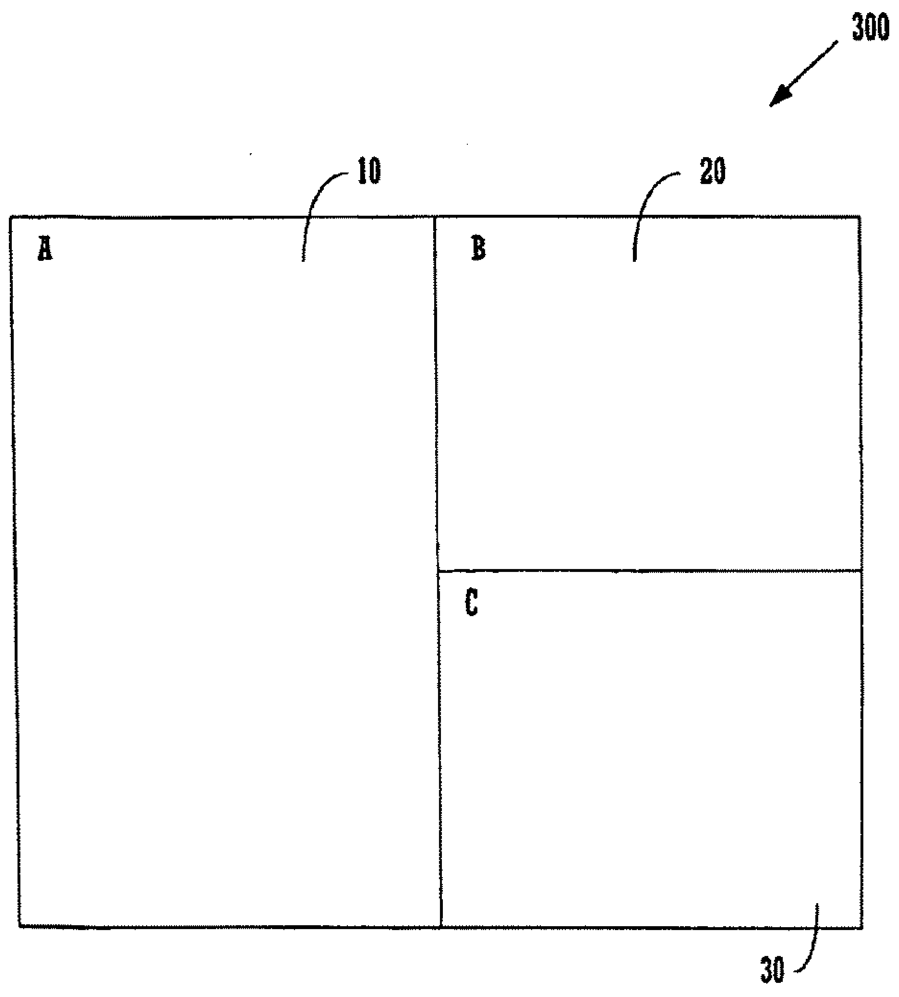


FIGURE 6

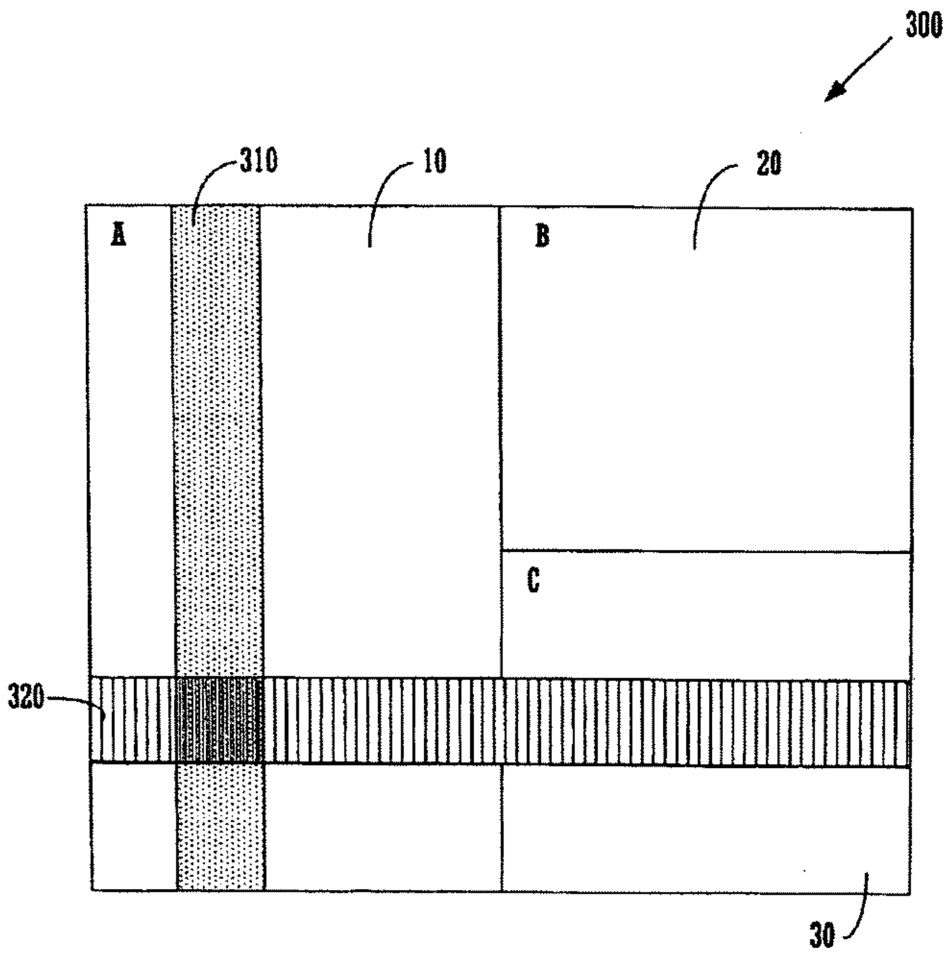


FIGURE 7

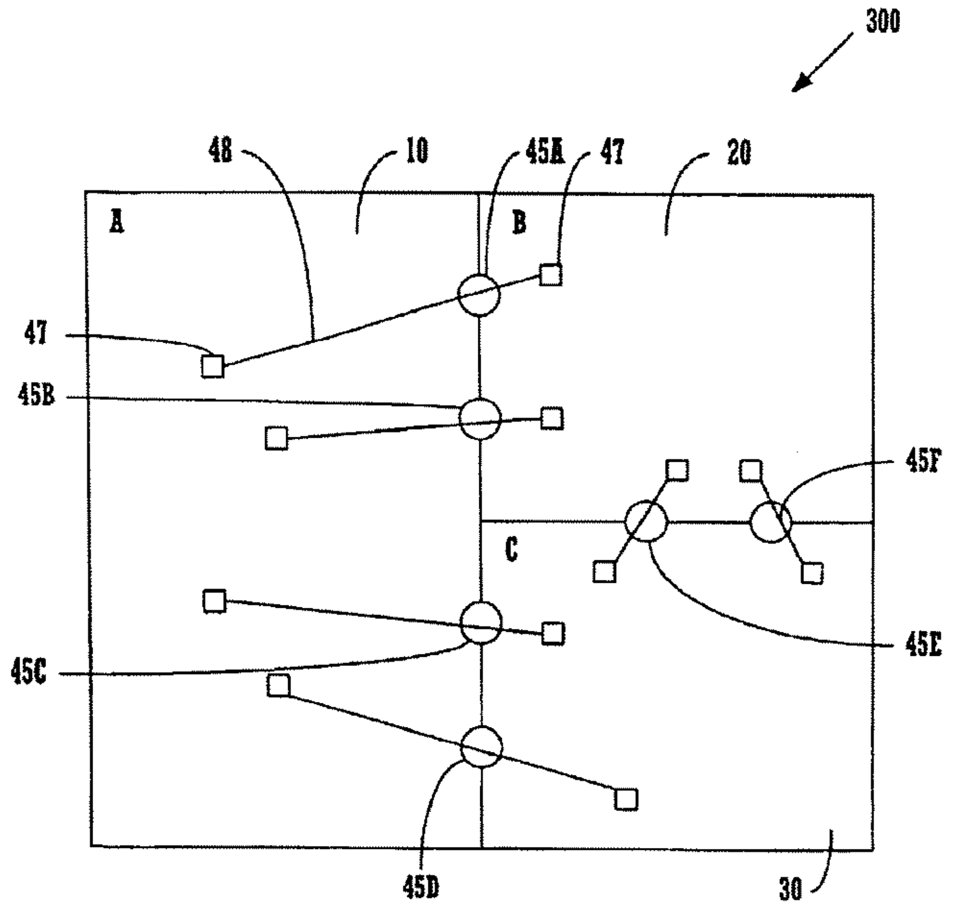


FIGURE 8

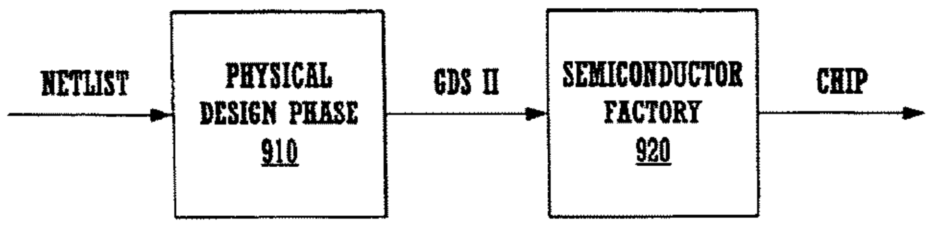


FIGURE 9A
(Prior Art)

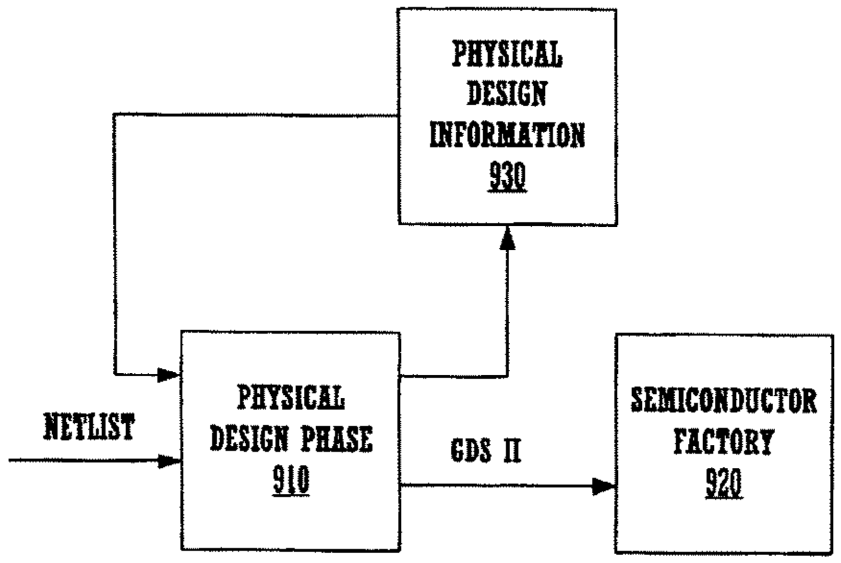

PHYSICAL DESIGN INFORMATION 930
NETLIST
PHYSICAL DESIGN PHASE 910
GDS II
SEMICONDUCTOR FACTORY 920

FIGURE 9B

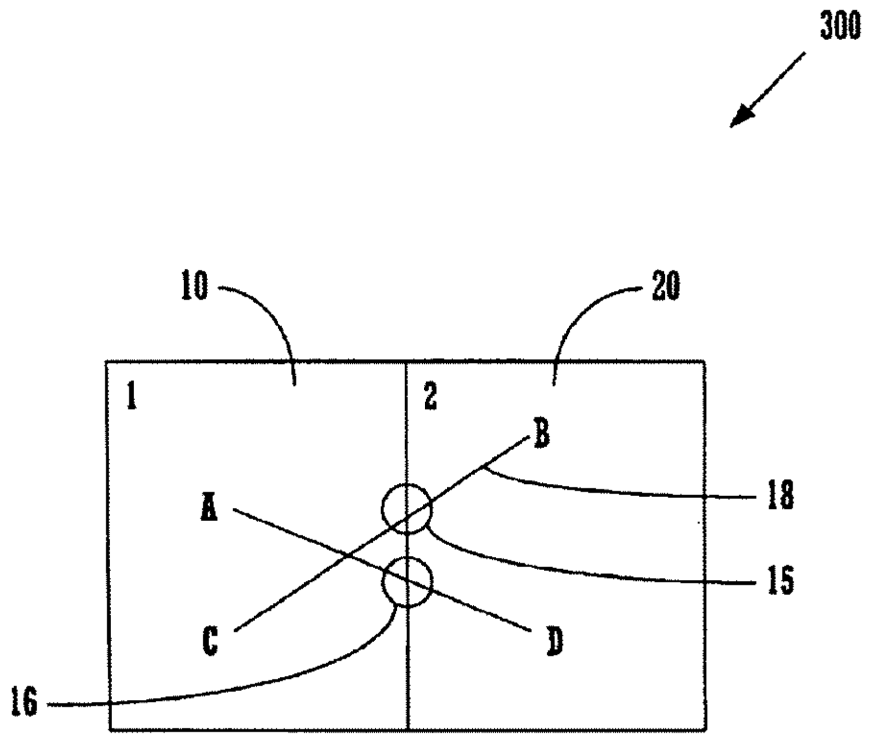


FIGURE 10A

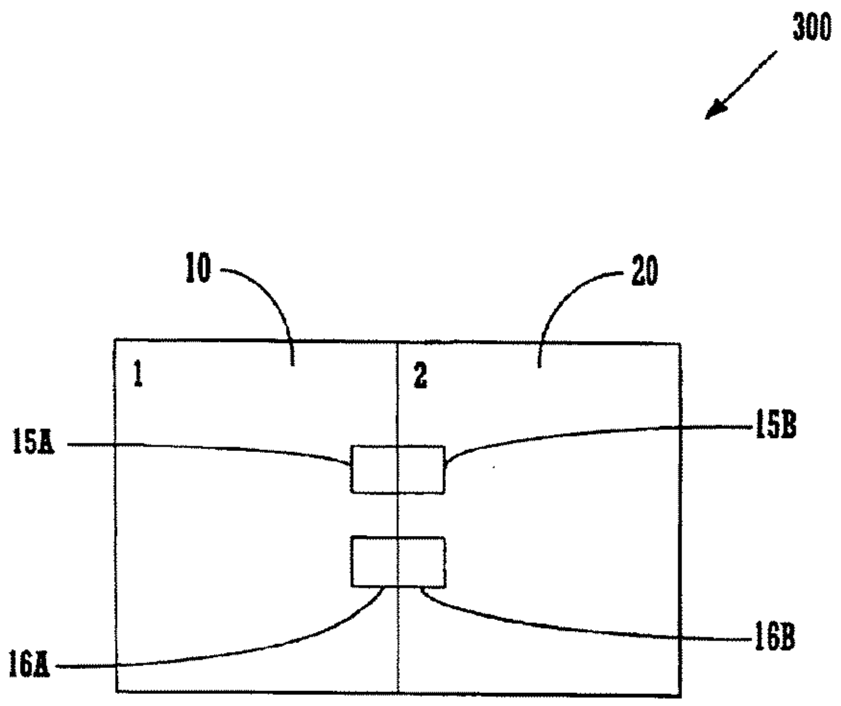


FIGURE 10B

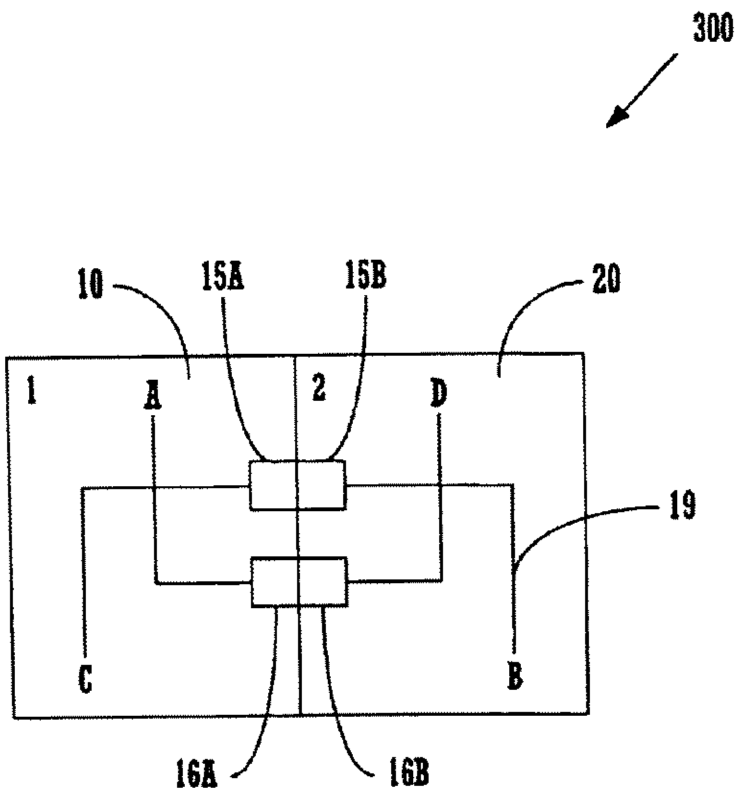


FIGURE 10C

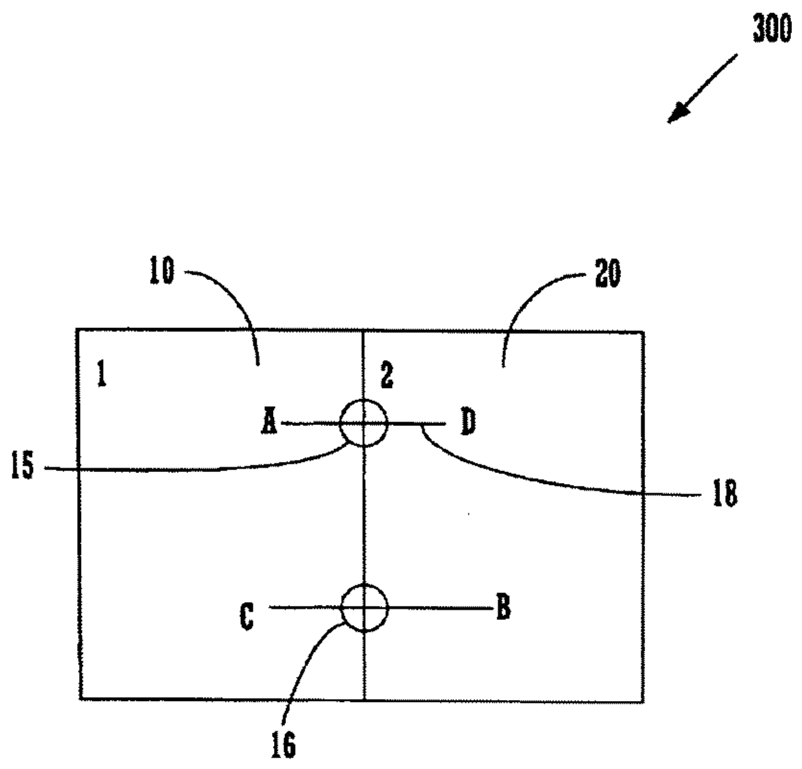


FIGURE 11A

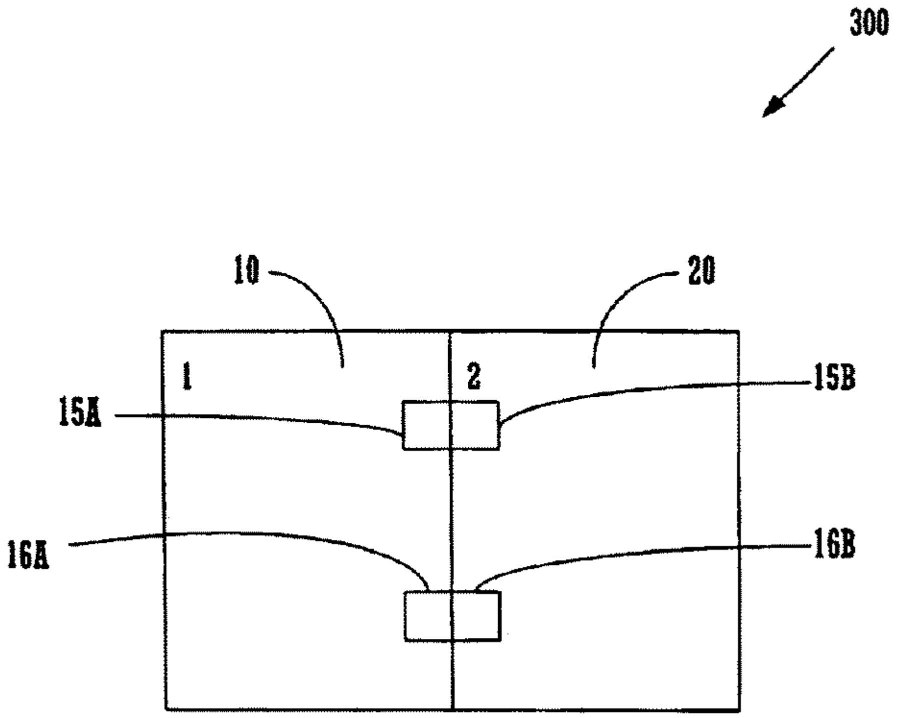


FIGURE 11B

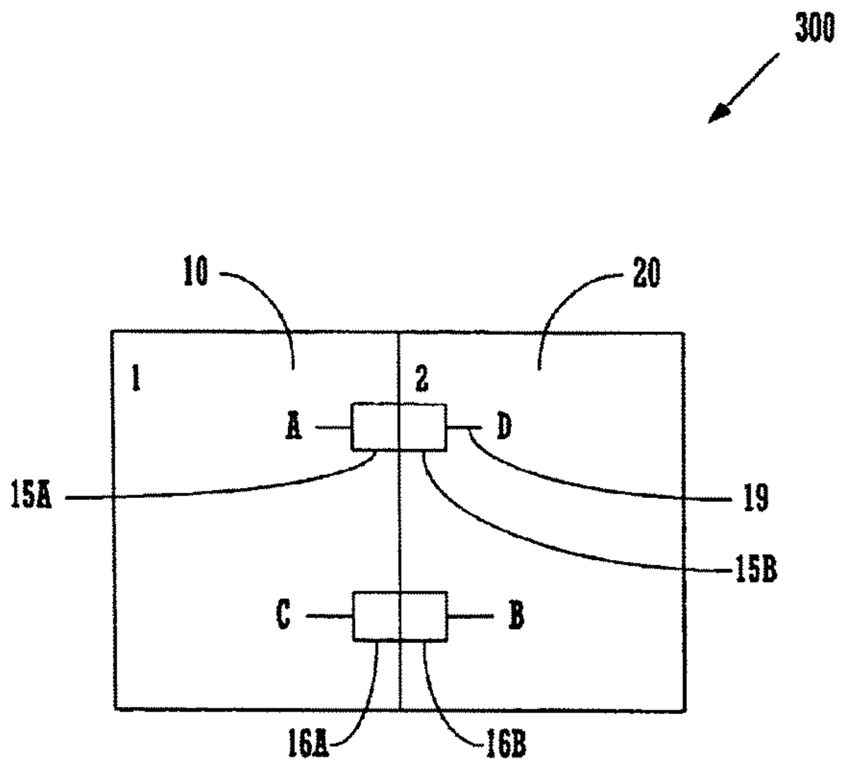


FIGURE 11C

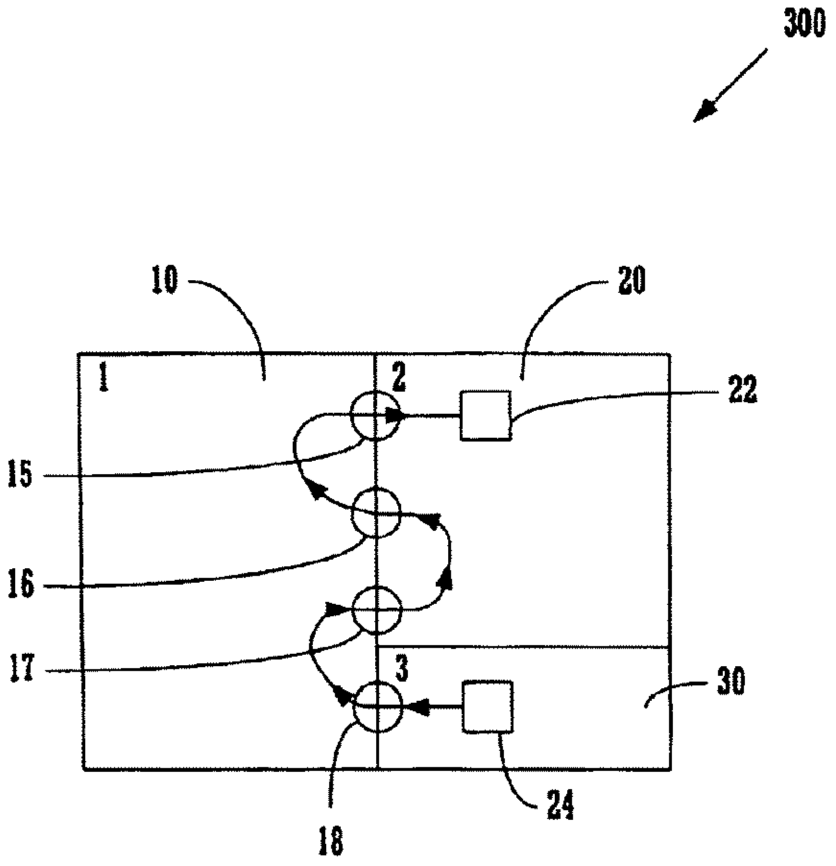


FIGURE 12A

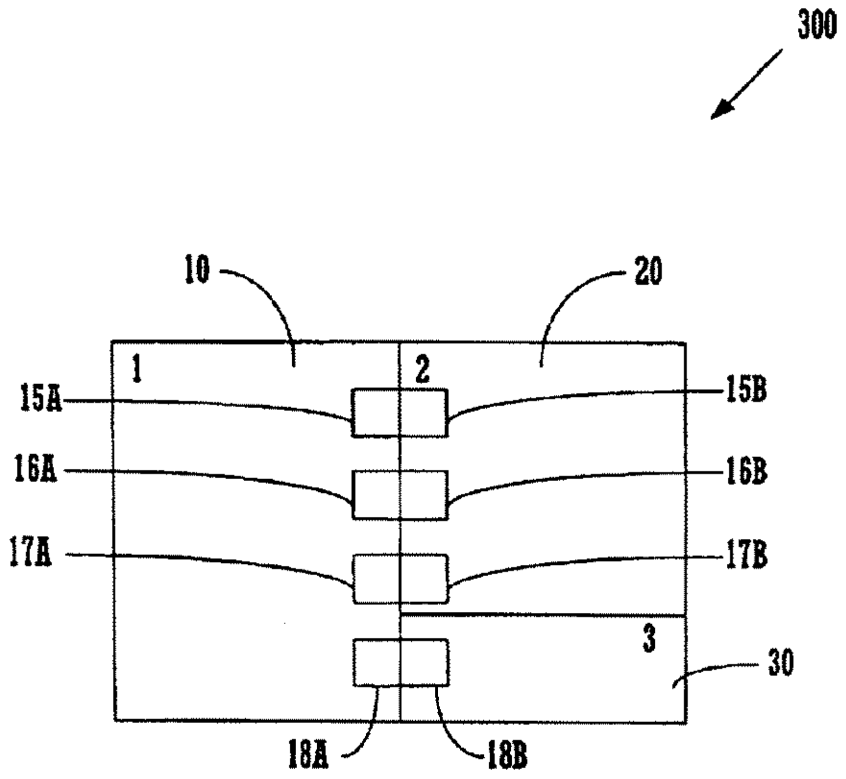


FIGURE 12B

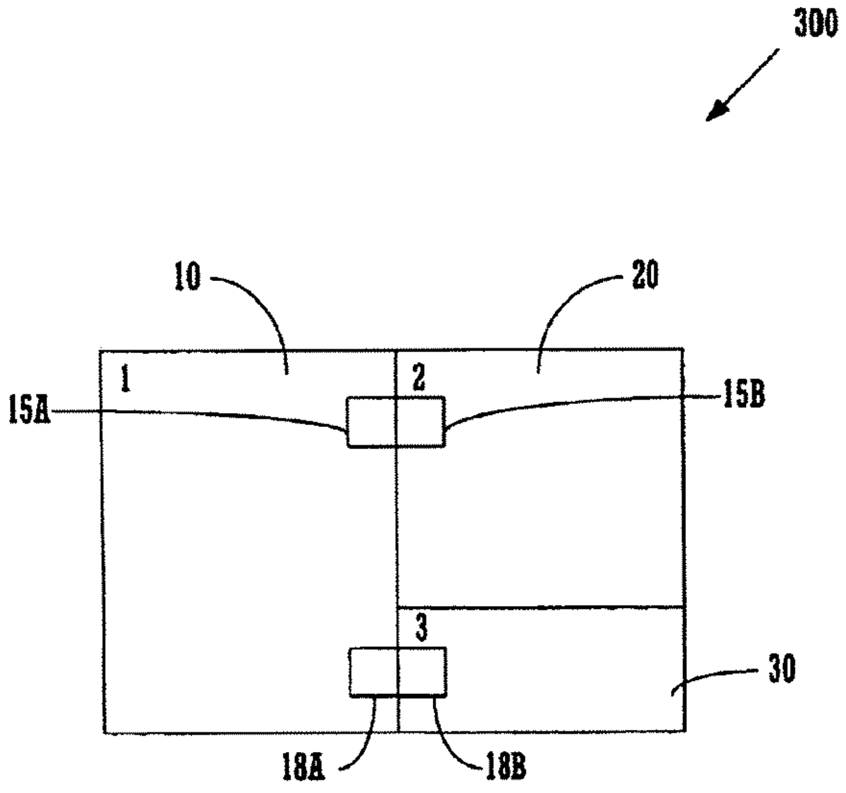


FIGURE 12C

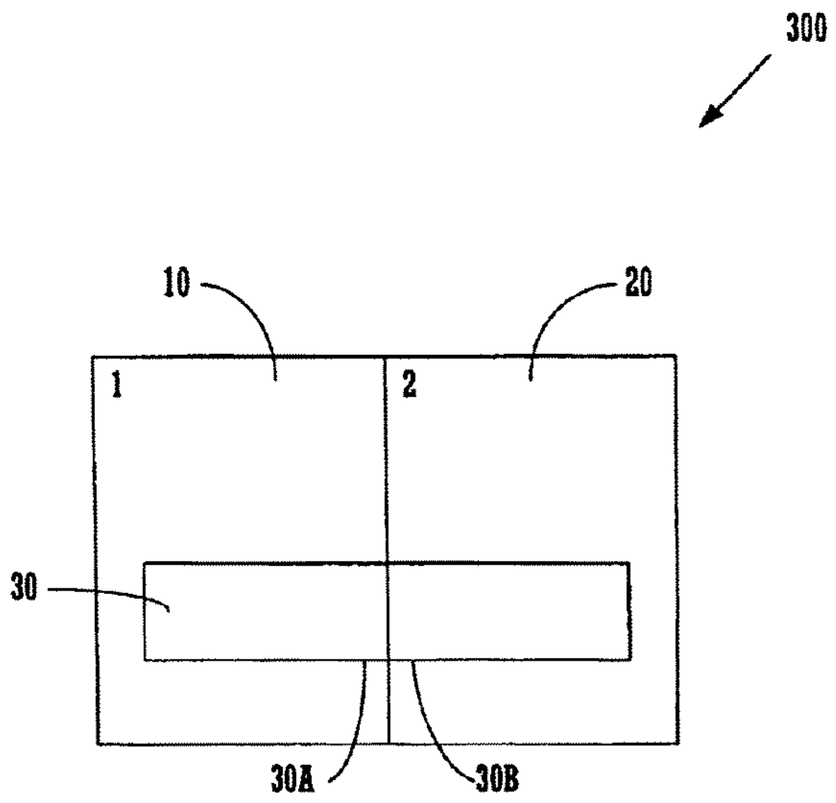


FIGURE 13A

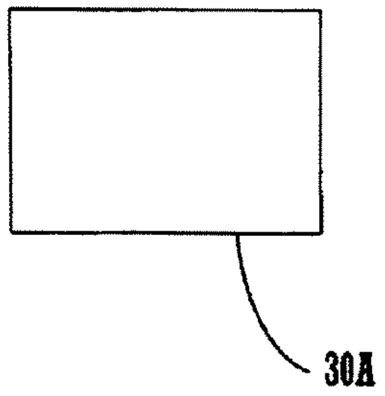


FIGURE 13B

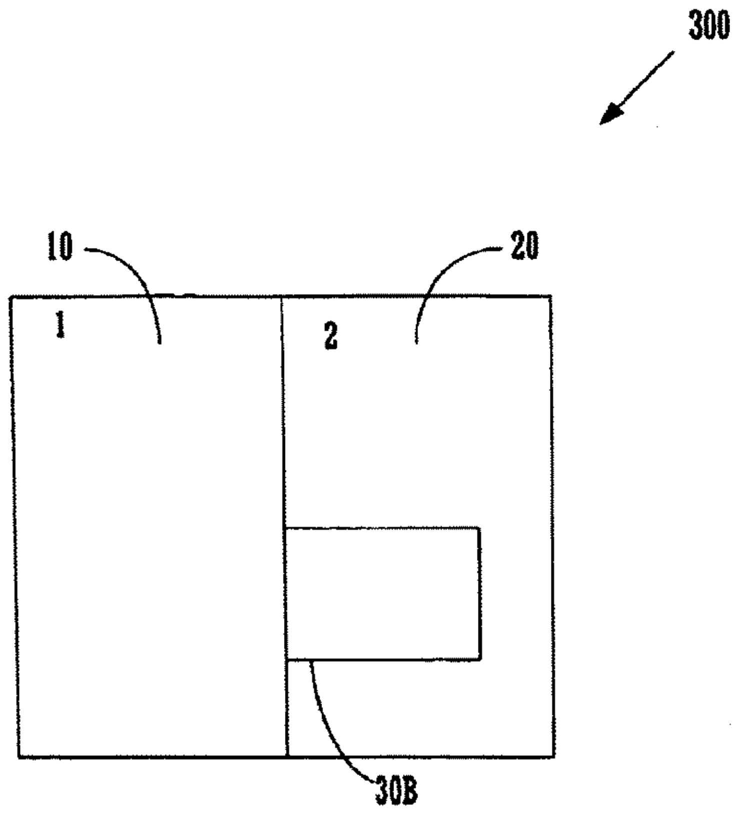


FIGURE 13C

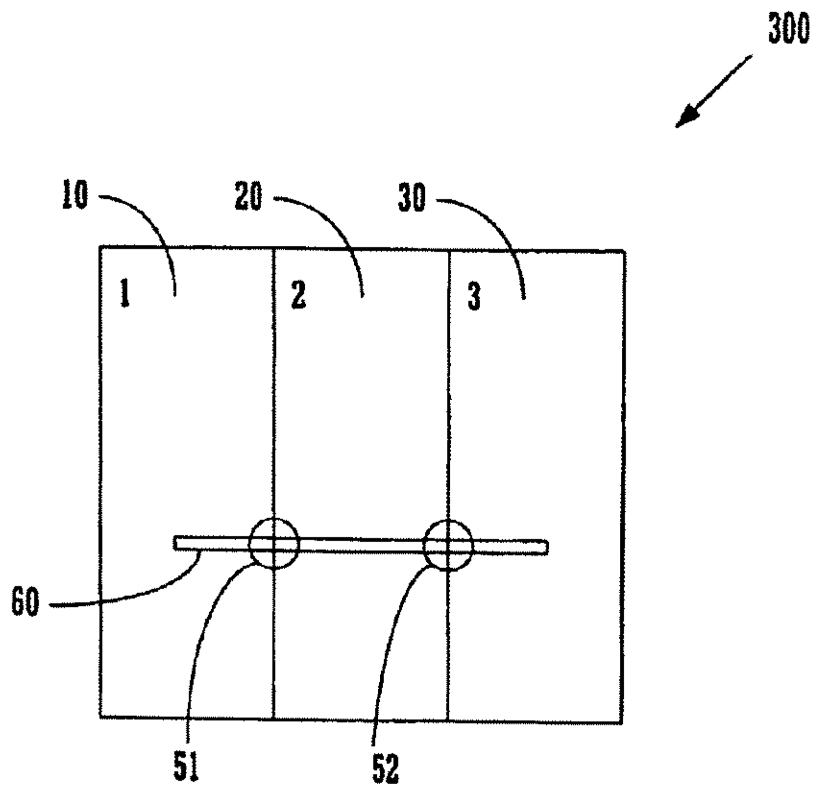


FIGURE 14A

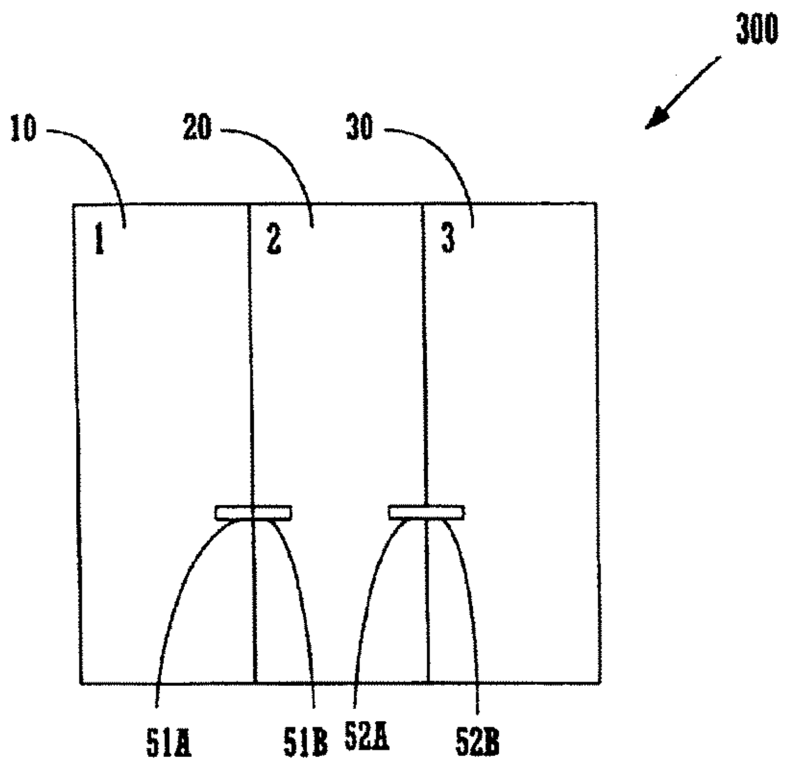


FIGURE 14B

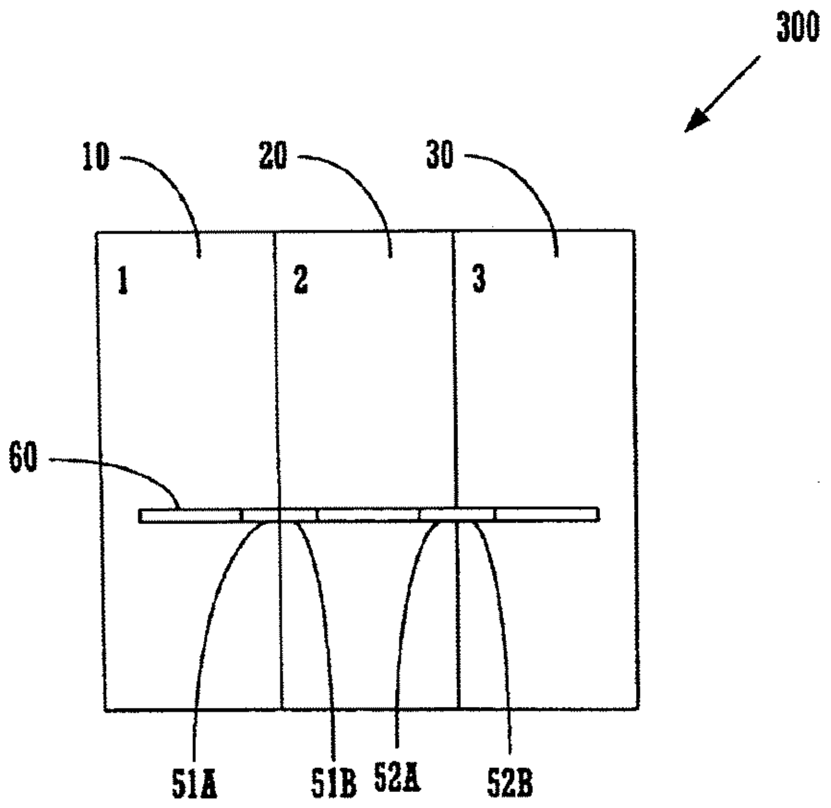


FIGURE 14C

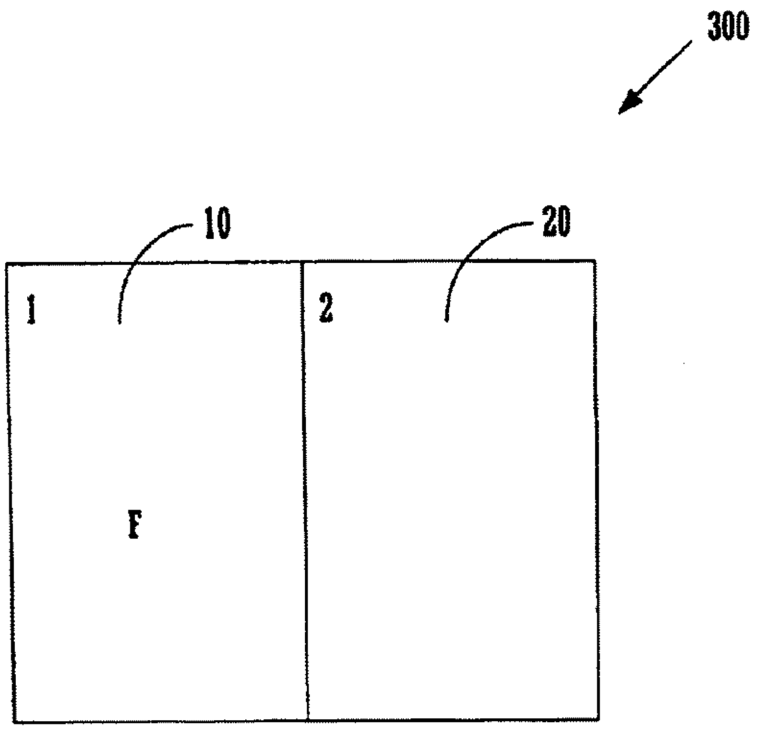


FIGURE 15A

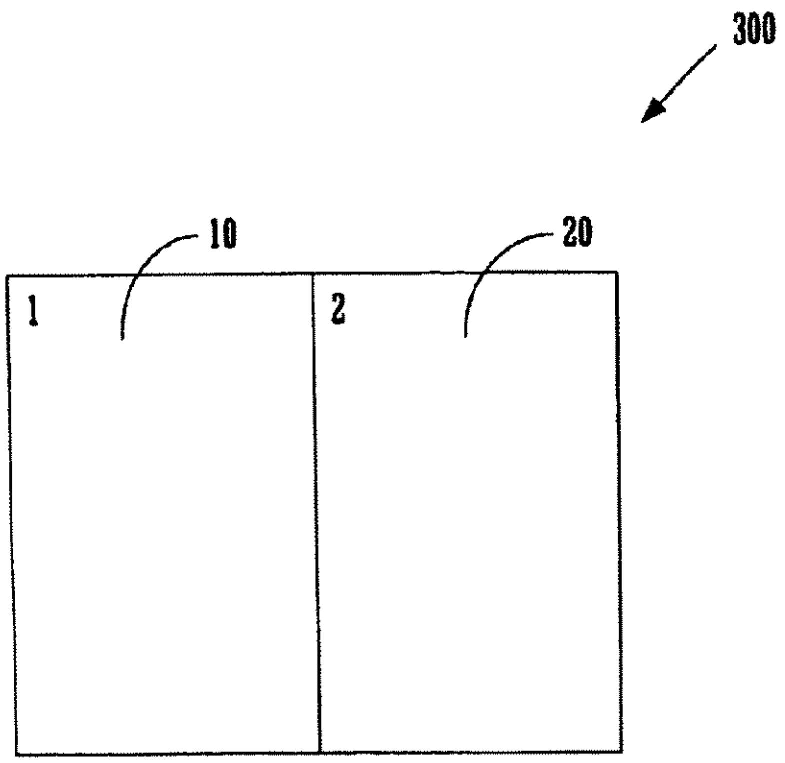


FIGURE 15B

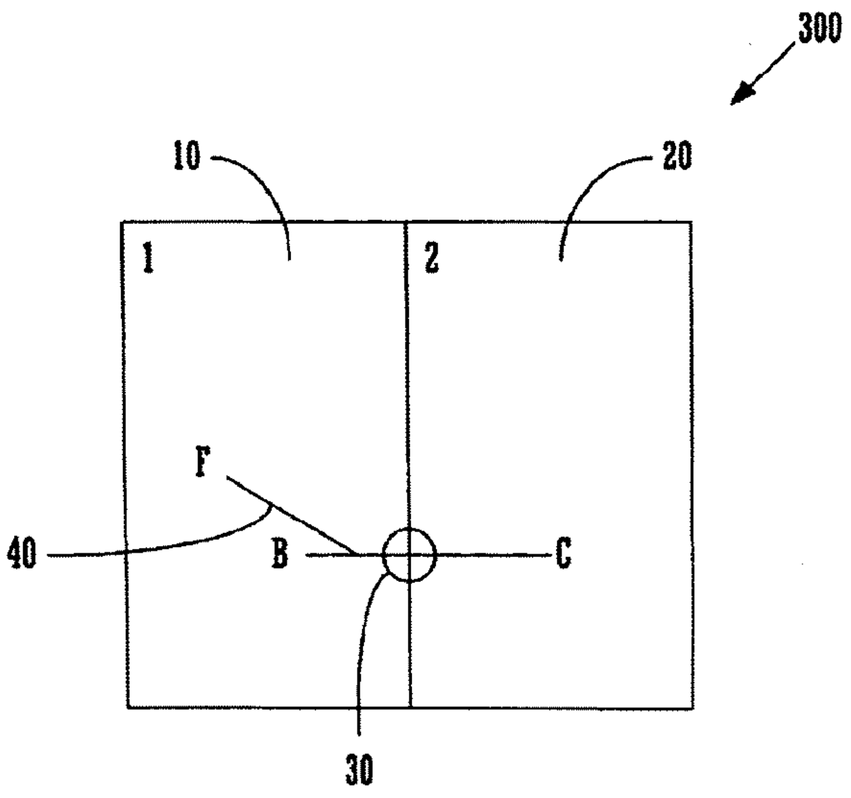


FIGURE 16A

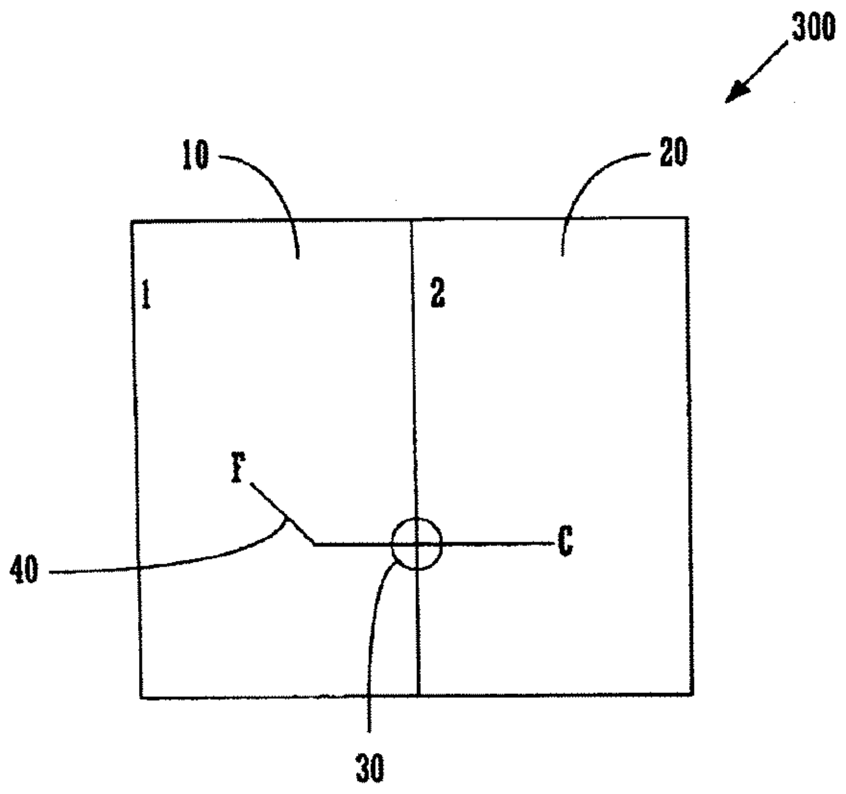


FIGURE 16B

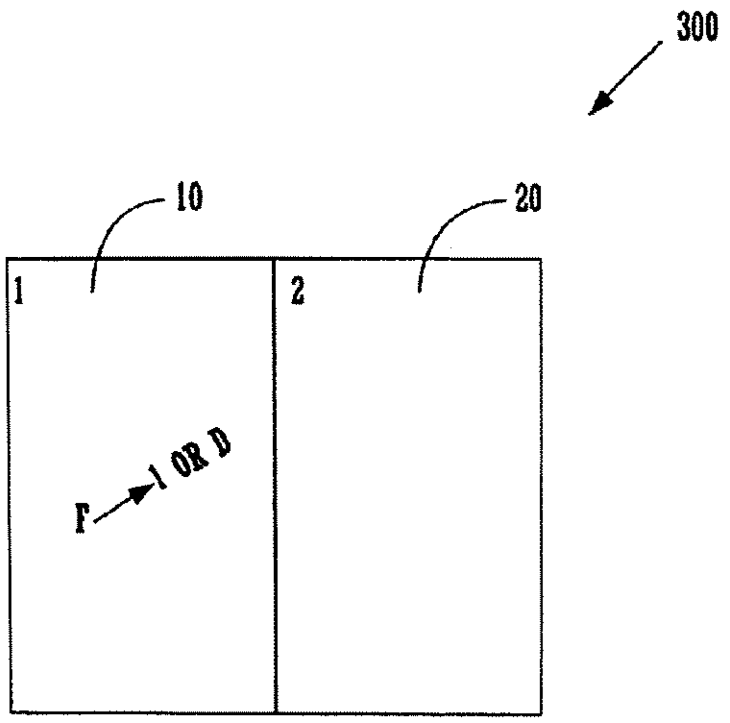


FIGURE 17A

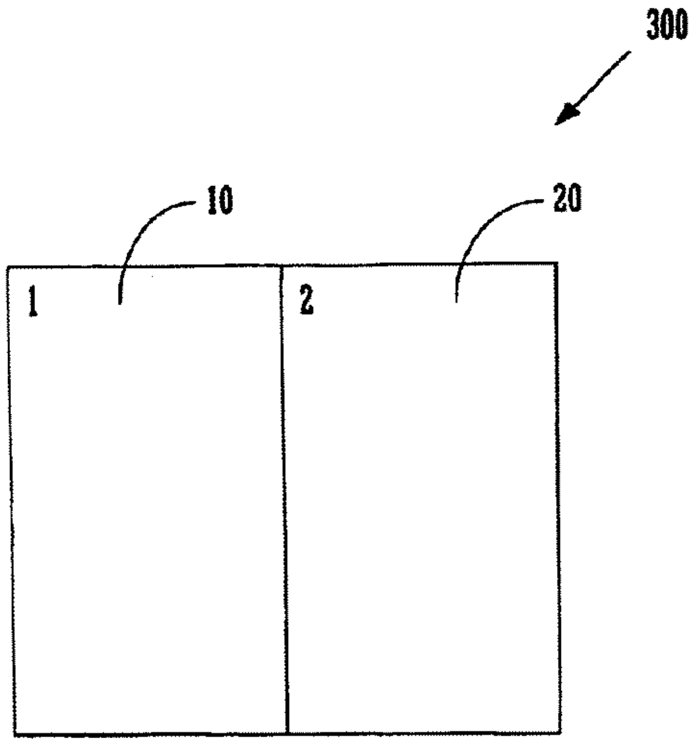


FIGURE 17B

OPTIMIZATION OF ABUTTED-PIN HIERARCHICAL PHYSICAL DESIGN

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention generally relates to the field of integrated circuit design. More particularly, the present invention relates to the field of software tools for hierarchical physical design.

2. Related Art

The tremendous advances in technology have been fueled by improvements in integrated circuit design. In particular, integrated circuits have become smaller and more complex. Integrated circuit design engineers depend on electronic design automation (EDA) software tools to facilitate the design of integrated circuits.

Typically, the integrated circuit design process begins with a specification which describes the functionality of the integrated circuit and may include a variety of constraints. Then, during a logic design phase, the logical implementation of the integrated circuit is determined. Several operations are performed to obtain a logical representation of the integrated circuit. Generally, EDA software tools use register transfer logic (RTL) to represent the integrated circuit. However, additional EDA software tools may be used.

After completing the logic design phase, the integrated circuit undergoes a physical design phase. Typically, the output of the logic design phase, is a netlist, which is then used in the physical design phase. Here, EDA software tools layout the integrated circuit to obtain a representation of the physical components in the integrated circuit, whereas the representation indicates the manner in which the integrated circuit will be implemented on a semiconductor chip. A variety of operations are performed on the layout of the integrated circuit.

At the end of the physical design phase, the representation of the semiconductor chip (in which the integrated circuit is implemented) is sent to a semiconductor manufacturing plant.

Typically, in the physical design phase, EDA software tools implement a flat physical design. For example, the components (standard cells, macrocells, etc.) of the integrated circuit are placed during a placement operation and are routed during a routing operation. However, as the integrated circuit becomes more complex, the EDA software tools struggle to perform the placement operation and the routing operation. In particular, the performance of the EDA software tools degrades since the EDA software tools have to manipulate very large files during the placement operation and the routing operation. Moreover, as the complexity of the integrated circuit increases, the time necessary to complete the physical design phase increases significantly.

Traditional hierarchical physical design has emerged as an alternative to the flat physical design. FIG. 1 illustrates the traditional hierarchical physical design **100**. Here, the components of the integrated circuit are partitioned into a plurality of blocks **10–30**. Each block **10–30** includes a plurality of pins **50**, whereas each pin **50** represents a location where a signal can enter the block **10–30** or a location where a signal can exit the block **10–30**. As illustrated in FIG. **1**, the traditional hierarchical physical design **100** includes a channel **40**. The channel **40** provides space in order to connect the pins **50** of the blocks **10–30** to one another via metal (not shown) or any other wiring material. The traditional hierarchical physical design **100** enables the placement operation and the routing operation (as well as other operations) for the blocks **10–30** to be performed in parallel with EDA software tools, reducing the time period of the physical design phase. Moreover, the performance of the EDA software tools is improved because the file for each block **10–30** is much smaller than the file for the entire integrated circuit of the flat physical design. More importantly, the EDA software tools are better suited to optimize each block **10–30** than to optimize the entire integrated circuit of the flat physical design. However, the traditional hierarchical physical design **100** generates wasted space in the channel **40** and generates wiring problems in the channel **40**, such as congestion and crosstalk. Moreover, the traditional hierarchical physical design **100** places and routes components at a top-level (shown in FIG. **1**) and a block-level (within each block **10–30**), causing inefficiencies and causing problems with EDA software tools which are configured to operate with flat physical designs.

SUMMARY OF THE INVENTION

An abutted-pin hierarchical physical design process is described. The abutted-pin hierarchical physical design provides solutions to the problems of the traditional hierarchical physical design and provides additional advantages and benefits. In particular, the abutted-pin hierarchical physical design does not have channels. Moreover, in the abutted-pin hierarchical physical design, components of the top-level are merged into the block-level so that the top-level netlist is reduced significantly.

In the integrated circuit design flow according to an embodiment of the present invention, the physical design phase receives the netlist from the logic design phase. In addition, the physical design phase receives physical design information, whereas the physical design information can be any information about a prior integrated circuit that has undergone the physical design phase. In an embodiment, the physical design information is stored in a database.

In an embodiment of the present invention, the integrated circuit design flow of the present invention is utilized to optimize pin assignment. In an embodiment of the present invention, excess pins formed along a boundary between two blocks are removed.

In an embodiment of the present invention, a software tool that performs a “press” operation preserves the properties associated with a segment of a top-level shape despite the shape operation (e.g., AND) being performed with the block and the top-level shape to obtain the segment.

If the top-level object has the press property, the top-level object retains its location when the top-level object is “pressed” into a block. If the top-level object does not have the press property, the top-level object generally does not retain its location when the top-level object is “pressed” into the block.

If in the top-level netlist, the instantiation of a block includes a port that is unused, (thus, not needed for the top-level routing for pin assignment), a software tool removes the port from the top-level netlist, but the block-level netlist of the block remains unchanged.

Some software tools are not able to represent the relationship that more than one port is coupled to a pin. Hence, a software tool removes one of the ports from the netlist based on some criteria, such as whether a port is an input port or an output port.

If in the top-level netlist, the instantiation of the block includes a port that is tied to either the power line (1) or the

ground line (0) rather to a port of another block, a software tool removes the port from the top-level netlist to avoid routing the port at the top-level. Moreover, the software tool ties the port to either the power line (1) or the ground line (0) in the block-level netlist of the block.

In an embodiment, a software tool performs an unwinding operation which adds to the block-level netlist—of bonding pad blocks—the ports (which were removed earlier by the software tool) that couple to the top-level inputs and to the top-level outputs. Thus, the netlist modified by the physical design phase (e.g., repeater and buffers are added to the netlist) can be compared with the netlist originally received from the logic design phase. In particular, formal verification, layout versus schematic (LVS) verification, and design rules check (DRC) verification can be performed by software tools.

In an embodiment, each block-level netlist is partitioned into a first netlist and a second netlist. The second netlist and its associated extraction file of each block and the top-level netlist and its associated extraction file are utilized by software tools to perform the timing analysis. This timing analysis can be performed significantly faster than the case where the block-level netlist is not partitioned into the first netlist and the second netlist. In an embodiment, the timing graph resulting from the timing analysis can be analyzed to extract timing constraints (relating to the delay that can be generated by a block) for each block. Hence, if a block is optimized to meet its extracted timing constrains, the block is more likely to meet its timing parameter when the block interacts with the other blocks in the integrated circuit.

These and other advantages of the present invention will no doubt become apparent to those of ordinary skill in the art after having read the following detailed description of the preferred embodiments which are illustrated in the drawing figures.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and form a part of this specification, illustrate embodiments of the invention and, together with the description, serve to explain the principles of the present invention.

FIG. 1 illustrates the traditional hierarchical physical design **100**.

FIG. 2 illustrates an exemplary computer system **200** on which embodiments of the present invention may be practiced.

FIG. 3 illustrates an integrated circuit **300** generated with software tools according to an embodiment of the abutted-pin hierarchical physical design process of the present invention.

FIG. 4 illustrates the abutted-pin hierarchical physical design process **400** according to an embodiment of the present invention.

FIG. 5 illustrates the abutted-pin hierarchical physical design process **500** as performed at the block-level in a particular block (**450A–450C** of FIG. 4) after step **440** of FIG. 4.

FIG. 6 illustrates the layout of the blocks **10–30** is established.

FIG. 7 illustrates a clock wire **320** and a power wire **310** of the top-level.

FIG. 8 illustrates a top-level route for obtaining the pin assignments for each block **10–30**.

FIG. 9A illustrates the integrated circuit design flow of the prior art.

FIG. 9B illustrates the integrated circuit design flow according to an embodiment of the present invention.

FIG. 10A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention and using the integrated circuit design flow of the prior art (FIG. 9A), showing the top-level routing for pin assignment.

FIG. 10B illustrates the integrated circuit **300** of FIG. 10A at the block-level.

FIG. 10C illustrates the integrated circuit **300** of FIG. 10B at the block-level.

FIG. 11A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention and using the integrated circuit design flow of the present invention (FIG. 9B), showing the top-level routing for pin assignment.

FIG. 11B illustrates the integrated circuit **300** of FIG. 11A at the block-level.

FIG. 11C illustrates the integrated circuit **300** of FIG. 11B at the block-level.

FIG. 12A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention, showing the top-level routing for pin assignment.

FIG. 12B illustrates the integrated circuit **300** of FIG. 12A at the block-level.

FIG. 12C illustrates the integrated circuit **300** of FIG. 12B, showing the removal of excess pins.

FIG. 13A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention, showing the top-level routing for a top-level object **30** (e.g., routing metal).

FIG. 13B illustrates the segment **30A** of FIG. 13A.

FIG. 13C illustrates the integrated circuit **300** of FIG. 13A in the top-level, showing that the segment **30A** has been removed from the top-level netlist and merged into the block-level netlist of block1 **10**.

FIG. 14A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention, showing the top-level routing for a top-level object **60** (e.g., routing metal).

FIG. 14B illustrates the integrated circuit **300** at the block-level.

FIG. 14C illustrates the integrated circuit **300** at the block-level.

FIG. 15A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention, showing the top-level routing for pin assignment.

FIG. 15B illustrates that the port F of block1 **10** has been removed from the top-level netlist.

FIG. 16A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention, showing the top-level routing for pin assignment.

FIG. 16B illustrates that the port B of block1 **10** has been removed from the netlist for the top-level routing for pin assignment.

FIG. 17A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention, showing the top-level routing for pin assignment.

FIG. 17B illustrates that the port F of block1 **10** has been removed from the top-level netlist.

The drawings referred to in this description should not be understood as being drawn to scale except if specifically noted.

DETAILED DESCRIPTION OF THE INVENTION

Reference will now be made in detail to the preferred embodiments of the present invention, examples of which are illustrated in the accompanying drawings. While the invention will be described in conjunction with the preferred embodiments, it will be understood that they are not intended to limit the invention to these embodiments. On the contrary, the invention is intended to cover alternatives, modifications and equivalents, which may be included within the spirit and scope of the invention as defined by the appended claims. Furthermore, in the following detailed description of the present invention, numerous specific details are set forth in order to provide a thorough understanding of the present invention. However, it will be recognized by one of ordinary skill in the art that the present invention may be practiced without these specific details. In other instances, well known methods, procedures, components, and circuits have not been described in detail as not to unnecessarily obscure aspects of the present invention.

Notation and Nomenclature

Some portions of the detailed descriptions which follow are presented in terms of procedures, logic blocks, processing, and other symbolic representations of operations on data bits within a computer memory. These descriptions and representations are the means used by those skilled in the data processing arts to most effectively convey the substance of their work to others skilled in the art. In the present application, a procedure, logic block, process, etc., is conceived to be a self-consistent sequence of steps or instructions leading to a desired result. The steps are those requiring physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated in a computer system. It has proved convenient at times, principally for reasons of common usage, to refer to these signals a s bits, values, elements, symbols, characters, terms, numbers, or the like.

It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities. Unless specifically stated otherwise as apparent from the following discussions, it is appreciated that throughout the present invention, a variety of terms are discussed that refer to the actions and processes of an electronic system or a computer system, or other electronic computing device/system. The computer system or similar electronic computing device manipulates and transforms data represented as physical (electronic) quantities within the computer system's registers and memories into other data similarly represented as physical quantities within the computer system memories or registers or other such information storage, transmission, or display devices. The present invention is also well suited to the use of other computer systems such as, for example, optical , mechanical, or quantum computers.

Exemplary Computer System Environment

Aspects of the present invention are discussed in terms of steps executed on a computer system. Although a variety of different computer systems can be used with the present invention, an exemplary computer system **200** is shown in FIG. **2**.

With reference to FIG. **2**, portions of the present invention are comprised of computer-readable and computer executable instructions which reside, for example, in computer-usable media of an electronic system such as the exemplary computer system. FIG. **2** illustrates an exemplary computer system **200** on which embodiments of the present invention may be practiced. It is appreciated that the computer system **200** of FIG. **2** is exemplary only and that the present invention can operate within a number of different computer systems including general-purpose computer systems and embedded computer systems.

Computer system **200** includes an address/data bus **110** for communicating information, a central processor **101** coupled with bus **110** for processing information and instructions, a volatile memory **102**. (e.g., random access memory RAM) coupled with the bus **110** for storing information and instructions for the central processor **101** and a non-volatile memory **103** (e.g., read only memory ROM) coupled with the bus **110** for storing static information and instructions for the processor **101**. Exemplary computer system **200** also includes a data storage device **104** ("disk subsystem") such as a magnetic or optical disk and disk drive coupled with the bus **110** for storing information and instructions. Data storage device **104** can include one or more removable magnetic or optical storage media (e.g., diskettes, tapes) which are computer readable memories. Memory units of computer system **200** include volatile memory **102**, non-volatile memory **103** and data storage device **104**.

Exemplary computer system **200** can further include an optional signal generating device **108** (e.g., a network interface card "NIC") coupled to the bus **110** for interfacing with other computer systems. Also included in exemplary computer system **200** of FIG. **2** is an optional alphanumeric input device **106** including alphanumeric and function keys coupled to the bus **110** for communicating information and command selections to the central processor **101**. Exemplary computer system **200** also includes an optional cursor control or directing device **107** coupled to the bus **110** for communicating user input information and command selections to the central processor **101**. An optional display device **105** can also be coupled to the bus **110** for displaying information to the computer user. Display device **105** may be a liquid crystal device, other flat panel display, cathode ray tube, or other display device suitable for creating graphic images and alphanumeric characters recognizable to the user. Cursor control device **107** allows the user to dynamically signal the two-dimensional movement of a visible symbol (cursor) on a display screen of display device **105**. Many implementations of cursor control device **107** are known in the art including a trackball, mouse, touch pad, joystick or special keys on alphanumeric input device **106** capable of signaling movement of a given direction or manner of displacement. Alternatively, it will be appreciated that a cursor can be directed and/or activated via input from alphanumeric input device **106** using special keys and key sequence commands.

Abutted-pin Hierarchical Physical Design

FIG. **3** illustrates an integrated circuit **300** generated with software tools according to the abutted-pin hierarchical physical design process of the present invention. The abutted-pin hierarchical physical design provides solutions to the problems of the traditional hierarchical physical design (see FIG. 1) and provides additional advantages and benefits. In particular, the abutted-pin hierarchical physical design does not have channels. Moreover, in the abutted-pin hierarchical physical design, components of the top-level are merged into the block-level so that the top-level netlist is reduced to instantiations of each block **10–30** and **60–94**.

As illustrated in FIG. **3**, the abutted-pin hierarchical physical design **300** includes a plurality of blocks **10–30** and **60–94**. The netlist of the integrated circuit **300** is partitioned into the plurality of blocks **10–30** and **60–94** such that each block **10–30** and **60–94** has a block level netlist. Blocks **10–30** have the major or core components of the integrated circuit **300**. Blocks **60–94** have the bonding pads and other support circuitry of the integrated circuit **300**. The blocks **10–30** and **60–94** can be rectangular in shape and can be rectilinear in shape. It should be understood that the integrated circuit **300** can have any number of blocks.

Each block **10–30** and **60–94** has one or more pins **50**, whereas each pin **50** represents a location where a signal can enter the block **10–30** and **60–94** or a location where a signal can exit the block **10–30** and **60–94**. The edge or boundary of each block **10–30** and **60–94** rests against the edge or boundary of another block **10–30** and **60–94**, such that the pin **50** of one block abuts the pin **50** of another block.

Moreover, the top-level components or objects (e.g., timing components, clock distribution wiring, power distribution wiring, repeaters, buffers, etc.) are not visible because they have been merged into the blocks **10–30** and **60–94** by a "press" operation performed by a software tool. First, the top-level objects (e.g., timing components, clock distribution wiring, power distribution wiring, repeaters, buffers, etc.) are placed and routed at the top-level (the top-level is shown in FIG. **3**). In the "press" operation, the top-level objects (e.g., timing components, clock distribution wiring, power distribution wiring, repeaters, buffers, etc.) that are within the boundary of a block **10–30** and **60–94** are removed from the top-level netlist and merged into the block-level netlist of that block **10–30** and **60–94**. Hence, the abutted-pin hierarchical physical design **300** can be optimized by separately optimizing the individual blocks **10–30** and **60–94**. Thus, the software tools can generate (e.g., perform placement, routing, timing, verification, etc.) and optimize the individual blocks **10–30** and **60–94** in parallel. Moreover, a bug within an individual block **10–30** and **60–94** can be corrected by returning that individual block to the logic design phase, while the other blocks continue to undergo the physical design phase.

FIG. **4** illustrates the abutted-pin hierarchical physical design process **400** according to an embodiment of the present invention. At **410**, a software tool receives the netlist of the integrated circuit from the logic design phase, as described above. The netlist is partitioned into a plurality of blocks, each block having a block-level netlist. In an embodiment, the partitioning of the netlist focuses on reducing the number of ports or terminals of a block that need to couple to the ports or terminals of other blocks.

At **420**, a software tool performs top-level floor planning. Here, the layout of each block is determined. At the end of the top-level floor planning, the top-level for an integrated circuit **300** (as shown in FIG. **6**) is generated. As illustrated in FIG. **6**, the layout of the blocks **10–30** is established. In FIG. **6**, the bonding pads **60–94** (of FIG. **3**) have been omitted.

At **430**, software tools perform top-level placement and routing for the top-level objects (e.g., timing components,

clock distribution wiring, power distribution wiring, repeaters, buffers, etc.). FIG. **7** illustrates a clock wire **320** and a power wire **310** of the top-level. The clock wire **320** is routed over BlockA **10** and BlockC **30**. The power wire **310** is routed over BlockA **10**. It should be understood that any number of additional top-level objects can be placed and routed at the top-level.

At **440**, a software tool performs a top-level route for obtaining the pin assignments for each block **10–30**, as illustrated in FIG. **8**. Since each block **10–30** has one or more ports or terminals **47** that needs to couple to a port or terminal of another block **10–30**, the pins for each block **10–30** have to be defined. Initially, the ports **47** of each block **10–30** are placed in a general random location within each block at the top-level since the actual location of the port **47** is not known until a placement operation is performed at the block-level. As illustrated in FIG. **8**, the location **45A–45F** where a routing wire **48** crosses a boundary between two blocks is defined as a pin for each of the blocks **10–30**, facilitating creation of pins that are abutted. In an embodiment, a software tool creates each pin to have a width that is equivalent to the width of the routing wire **48** at the boundary between the two blocks. The pins **50** are illustrated in FIG. **3**.

At **450A–450C**, the abutted-pin hierarchical physical design process **400** enables software tools to generate and to optimize each block **10–30** in parallel at the block-level.

FIG. **5** illustrates the abutted-pin hierarchical physical design process **500** as performed at the block-level in a particular block (**450A–450C** of FIG. **4**) after step **440** of FIG. **4**.

At **510**, a software tool performs press operations. The top-level objects illustrated in FIG. **7** (e.g., a clock wire **320** and a power wire **310**) and which are located within the boundary of a particular block, are pressed into the particular block. In particular, the top-level objects that are within the boundary of a particular block are removed from the top-level netlist and merged into the block-level netlist of that particular block. Moreover, the pins for the particular block are generated based on the location where the routing wire crosses the boundary between two blocks, as illustrated in FIG. **8** and FIG. **3**.

At **520**, a software tool performs block-level floor planning for the particular block. At **530**, a software tool performs a block-level placement operation for the particular block. At **540**, software tools perform a variety of block-level operations to optimize the particular block. Additionally, at **550**, a block-level route is performed for the particular block by a software tool. At **552** and **554**, software tools perform a block-level extraction operation for determining capacitance and resistance at the nodes and perform block-level timing analysis operations for the particular block.

At **560** and **570**, a variety of software tools perform a number of verification operations such as formal verification, layout versus schematic (LVS) verification, and design rules check (DRC) verification.

FIG. **9A** illustrates the integrated circuit design flow of the prior art. As illustrated in FIG. **9A**, the physical design phase **910** receives the netlist from the logic design phase (not shown). The physical design phase **910** generates the physical design for the integrated circuit and outputs a GDS II file. The GDS II file is received by the semiconductor factory **920**. The integrated circuit is fabricated by the semiconductor factory **920** on a semiconductor chip.

FIG. **9B** illustrates the integrated circuit design flow according to an embodiment of the present invention. As illustrated in FIG. 9B, the physical design phase **910** receives the netlist from the logic design phase (not shown). In addition, the physical design phase **910** receives physical design information **930** , whereas the physical design information **930** can be any information about a prior integrated circuit that has undergone the physical design phase **910**. In an embodiment, the physical design information **930** is stored in a database. For example, the physical design information **930** can be pin assignments of the prior integrated circuit, optimal clock distribution tree of the prior integrated circuit, parasitic extraction data of the prior integrated circuit, locations of obstructions such as a RAM of the prior integrated circuit, identification of congested blocks of the prior integrated circuit, metal resources for the blocks of the prior integrated circuit, or any other information which can facilitate optimizing the current integrated circuit. Thus, the software tools of the physical design phase **910** can customize the current integrated circuit to avoid the problems of the prior integrated circuit and to realize the benefits of the prior integrated circuit.

In the physical design phase **910**, decisions made at the top-level with respect to the top-level objects, significantly influence the creation of problems at the block-level and the optimization operations at the block-level. By using physical design information **930** (concerning the block-level of the prior integrated circuit) at the top-level of the current integrated circuit, the decisions made at the top-level with respect to the top-level objects of the current integrated circuit will be able to reduce the problems present in the prior integrated circuit and will be able to generate solutions to overcome the problems present in the prior integrated circuit, improving the optimization of the abutted-pin hierarchical physical design process of the present invention. Thus, if the physical design information **930** has information about several prior integrated circuits, the current integrated circuit is more likely to be optimized.

In addition, the physical design phase **910** generates the physical design for the integrated circuit and outputs a GDS II file. Moreover, the physical design phase **910** stores physical design information **930** of the current integrated circuit to be used in the physical design phase **910** of a future integrated circuit. The GDS II file is received by the semiconductor factory **920**. The integrated circuit is fabricated by the semiconductor factory **920** on a semiconductor chip.

FIG. **10A** illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention and using the integrated circuit design flow of the prior art (FIG. 9A), showing the top-level routing for pin assignment. The port C of block1 **10** is routed to port B of block2 **20**. The port A of block1 **10** is routed to port D of block2 **20**. This top-level routing has been performed after ports A–D where placed in a generally random location within each block **10–20** at the top-level since the actual locations of the ports A–D are not known until a placement operation is performed at the block-level. Here, the software tools at the top-level do not have access to the physical design information of a prior integrated circuit. The locations **15** and **16** are where the routing metal **18** crosses the boundary between two blocks **10** and **20**.

FIG. **10B** illustrates the integrated circuit **300** of FIG. **10A**. At the block-level, the pins **15A** and **16A** were formed for block1 **10**. At the block-level, the pins **15B** and **16B** were formed for block2 **20**, whereas pin **15A** abuts pin **15B** and pin **16A** abuts pin **16B**. The pins **15A** and **15B** were formed at location **15** of FIG. **10A**. The pins **16A** and **16B** were formed at location **16** of FIG. **10A**.

FIG. **10C** illustrates the integrated circuit **300** of FIG. **10B** at the block-level. As illustrated in FIG. **10C**, the block-level

placement operation for block1 **10** placed the ports A and C at locations that are different from the locations used to generate the pin assignments in FIG. **10A**. In addition, the block-level placement operation for block2 **20** placed the ports B and D at locations that are different from the locations used to generate the pin assignments in FIG. **10A**. Hence, the block-level routing operations for blocks **10** and **20** generated an inefficient amount of routing wire **19** to couple the ports to the pins in each block. In sum, the pin assignment affects the optimization of the routing wire **19**.

FIG. **11A** illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention and using the integrated circuit design flow of the present invention (FIG. 9B), showing the top-level routing for pin assignment. The port C of block1 **10** is routed to port B of block2 **20**. The port A of block1 **10** is routed to port D of block2 **20**. This top-level routing has been performed after each port A–D where placed in a particular location within each block **10–20** at the top-level, whereas the particular location was based on using the physical design information associated with the prior integrated circuit (FIGS. **10A–10C**). Here, the software tools at the top-level have access to the physical design information of the prior integrated circuit (FIGS. **10A–10C**). The locations **15** and **16** are where the routing metal **18** crosses the boundary between two blocks **10** and **20**.

FIG. **11B** illustrates the integrated circuit **300** of FIG. **11A** at the block-level. At the block-level, the pins **15A** and **16A** were formed for block1 **10**. At the block-level, the pins **15B** and **16B** were formed for block2 **20**, whereas pin **15A** abuts pin **15B** and pin **16A** abuts pin **16B**. The pins **15A** and **15B** were formed at location **15** of FIG. **1A**. The pins **16A** and **16B** were formed at location **16** of FIG. **11A**. Here, the pins **15A** and **15B** are associated with ports A and D, unlike FIG. **10B** where pins **15A** and **15B** were associated with ports C and B. Moreover, the pins **16A** and **16B** of FIG. **11B** are associated with ports C and B, unlike FIG. **10B** where pins **16A** and **16B** were associated with ports A and D.

FIG. **11C** illustrates the integrated circuit **300** of FIG. **11B** at the block-level. As illustrated in FIG. **11C**, the block-level placement operation for block1 **10** placed the ports A and C at locations that are different from the locations used to generate the pin assignments in FIG. **11A**. In addition, the block-level placement operation for block2 **20** placed the ports B and D at locations that are different from the locations used to generate the pin assignments in FIG. **11A**. However, the difference in the location of the ports between FIG. **11A** and FIG. **11C** is less than the difference in the location of the ports between FIG. **10A** and FIG. **10C**. Hence, the block-level routing operations for blocks **10** and **20** generated a more efficient amount of routing wire **19** to couple the ports to the pins in each block, compared to FIG. **10C**. In sum, the pin assignments generated with the use of the physical design information of the prior integrated circuit (FIGS. **10A–10C**) were more optimal than the pin assignments generated without the use of the physical design information of the prior integrated circuit (FIGS. **10A–10C**).

FIG. **12A** illustrates an integrated circuit **300** based on the abutted-pin a hierarchical physical design process of the present invention, showing the top-level routing for pin assignment. In the course of routing source port **24** of block3 **30** to destination port **22** of block2 **20**, the software tool that performs the top-level routing for pin assignment crosses the boundary between block1 **10** and block2 **20** at locations **15**,**16**, and **17**, whereas the locations **15**,**16**, and **17** will be defined as pins. The software tool is concerned with routing a path between the source port **24** and the destination port

22, but is not concerned about the number of times the path crosses the boundary between the same blocks.

FIG. 12B illustrates the integrated circuit **300** of FIG. 12A at the block-level. The pins **15A–15B**, **16A–16B**, and **17A–17B** are formed between block1 **10** and block2 **20**. The pins **18A–18B** are formed between block1 **10** and block3 **30**. The presence of pins **16A–16B** and **17A–17B** causes additional routing metal to be added to block1 **10** and block2 **20** so that pins **15A**, **16A**, and **17A** can be coupled within block1 **10** and so that pins **15B**, **16B**, and **17B** can be coupled within block2 **20**. Hence, one pair of pins (**15A–15B** or **16A–16B** or **17A–17B**) is sufficient.

FIG. 12C illustrates the integrated circuit **300** of FIG. 12B, showing the removal of excess pins. As illustrated in FIG. 120, excess pins **16A–16B** and **17A–17B** were removed from block1 **10** and block2 **20**. This removal is based on a plurality of criteria, such as the current flow direction between the source port **24** and the destination port **22**, the location of the excess pins relative to the source port **24** and the destination port **22**, or any other criteria. Here, the criteria kept pins **15A–15B** but deleted pins **16A–16B** and **17A–17B**.

FIG. 13A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention, showing the top-level routing for a top-level object **30** (e.g., routing metal). As described above, a software tool performs a press operation so that the portion of the top-level object **30** which is within the boundary of a particular block **10–20** is moved from the top-level netlist to the block-level netlist of the particular block **10–20**. In particular, the segment **30A** is pressed into block1 **10** while the segment **30B** is pressed into block2 **20**. In an embodiment, the shape operations of a database are utilized in performing the press operation. In FIG. 13A, an AND operation would be performed with block1 **10** and the shape **30** to obtain the segment **30A** (FIG. 13B). Typically, the routing metal **30** includes a plurality of properties that are stored in a database. These properties identify the routing metal **30** and describe the function of the routing metal **30**. However, in the shape operations (e.g., AND) of the prior art, the shape operation returns the segment **30A** (FIG. 13B) without its properties. Thus, these properties have to be reconstructed.

In the present invention, the software tool t hat performs the press operation preserves the properties associated with segment **30A** of the routing metal **30** despite the shape operation (e.g., AND) performed with block1 **10** and the shape **30** to obtain the segment **30A** (FIG. 13B).

FIG. 13C illustrates the integrated circuit **300** of FIG. 13A in the top-level, showing that the segment **30A** has been removed from the top-level netlist and merged into the block-level netlist of block1 **10**. Moreover, the properties associated with segment **30A** at the top level are transferred to the segment **30A** at the block-level.

FIG. 14A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention, showing the top-level routing for a top-level object **60** (e.g., routing metal). As illustrated in FIG. 14A, the top-level object **60** is routed through block1 **10**, block2 **20**, and block3 **30**. The locations **51–52** indicate top-level object **60** crosses a boundary between two blocks. In an embodiment, a press property is added to the properties of the top-level object **60** stored in a database. If the top-level object **60** has the press property, the top-level object **60** retains its location when the top-level object **60** is pressed into block1 **10**, block **20**, and block3 **30**, as illustrated in the block-level view of the integrated circuit **300** in FIG. 14C. If the top-level object **60** does not have the press property, the top-level object **60** generally does not retain its location when the top-level object **60** is pressed into block1 **10**, block2 **20**, and block3 **30**, as illustrated in the block-level view of the integrated circuit **300** in FIG. 14B. For example, top-level objects such as power and ground have the press property. As illustrated in FIG. 14B, the pins **51A–51B** and **52A–52B** are defined. However, the software tool is not constrained to placing the top-level object **60** in the block-level exactly as it was placed at the top-level. Moreover, the top-level object is placed in the block-level of block1 **10**, block2 **20**, and block3 **30** according to the separate placement and routing requirements of block1 **10**, block2 **20**, and block3 **30**. –FIG. 15A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention, showing the top-level routing for pin assignment. As illustrated in FIG. 15A, in the top-level netlist, the instantiation of block1 **10** includes port F that is unused, thus, not needed for the top-level outing for pin assignment. Hence, a software tool removes port F from the top-level netlist, but the block-level netlist of block1 **10** remains unchanged. In an embodiment, the software tool that performs the press operation removes the port F. FIG. 15B illustrates that the port F of block1 **10** has been removed from the top-level netlist.

FIG. 16A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention, showing the top-level routing for pin assignment. As illustrated in FIG. 16A, port F and port B of block1 **10** are coupled to port C of block2 **20** with a routing metal **40**. However, at location **30** the routing metal **40** crosses the boundary between block1 **10** and block2 **20**. If a pin is formed within block1 **10** at location **30**, the pin would be coupled to port F and to port B. However, some software tools are not able to represent this relationship (i.e., more than one port coupled to a pin). Hence, a software tool removes one of the ports (port F or port B) from the netlist based on some criteria, such as whether a port is an input port or an output port. FIG. 16B illustrates that the port B of block1 **10** has been removed from the netlist for the top-level routing for pin assignment.

FIG. 17A illustrates an integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention, showing the top-level routing for pin assignment. As illustrated in FIG. 17A, in the top-level netlist, the instantiation of block1 **10** includes a port F that is tied to either the power line (1) or the ground line (0) rather to a port of another block. Hence, a software tool removes port F from the top-level netlist to avoid routing the port F at the top-level. Moreover, the software tool ties the port F to either power line (1) or the ground line (0) in the block-level netlist of block1 **10**. FIG. 17B illustrates that the port F of block1 **10** has been removed from the top-level netlist.

As illustrated in FIG. 3, the integrated circuit **300** based on the abutted-pin hierarchical physical design process of the present invention includes a North bond pad block **60**, an East bond pad block **70**, a South bond pad block **80**, and a West bond pad block **90**, each having bond pad cells. The top-level netlist of the integrated circuit **300** includes one or more top-level inputs for receiving external signals and one or more top-level outputs for transmitting signals off the chip. The top-level inputs and the top-level outputs are coupled to bond pad cells. Typically, software tools which perform a routing operation are configured to not perform the routing operation if the netlist includes bond pad cells.

Since the North bond pad block **60**, the East bond pad block **70**, the South bond pad block **80**, and the West bond pad block **90** have bond pad cells in the block-level netlist, the software tools refuse to perform the routing operation in these blocks, preventing pins to be formed on the boundary between these blocks and the blocks **10–30** (the core blocks).

In the present invention, the bond pad cells are marked as macrocells rather than bond pad cells, allowing pins to be formed on the boundary between these blocks **60**, **70**, **80**, and **90** and the blocks **10–30** (the core blocks).

Typically, the block-level netlist of the North bond pad block **60**, the East bond pad block **70**, the South bond pad block **80**, and the West bond pad block **90** include nets to the top-level inputs and nets to the top-level outputs. Generally, the block-level netlist of the North bond pad block **60**, the East bond pad block **70**, the South bond pad block **80**, and the West bond pad block **90** include nets to the bond pad cells.

In an embodiment of the present invention, a software tool removes the nets to the top-level inputs and nets to the top-level outputs so that the physical design of the integrated circuit can be accomplished as described above. In an embodiment, the software tool removes in the block-level netlist the ports that couple to the top-level inputs and to the top-level outputs. Moreover, the software tool adds a property to the nets to the bond pad cells to indicate that these nets are suppose to couple to the top-level inputs and to the top-level outputs, facilitating an unwinding operation to re-establish at the block-level netlist the nets to the top-level inputs and nets to the top-level outputs that were removed earlier. The un winding operation adds to the block-level netlist the ports (which were removed earlier) that couple to the top-level inputs and to the top-level outputs. Thus, the netlist modified by the physical design phase (e.g., repeater and buffers are added to the netlist) can be compared with the netlist originally received from the logic design phase. In particular, formal verification, layout versus schematic (LVS) verification, and design rules check (DRC) verification can be performed by software tools.

A challenge with implementing an integrated circuit based on the abutted-pin hierarchical physical design process of the present invention involves analyzing the timing of signal paths that traverse more than one block. The timing of these global paths is difficult to analyzed compared to analyzing the timing of local paths, whereas local paths are signal paths that do not leave a block. One method of analyzing the timing of these global paths involves partitioning the block-level netlist of each block into a first netlist and a second netlist. The first netlist includes nets which start at a register (or flip-flop) and end at a register (or flip-flop) within the block, whereas each branch of the net also starts at a register (or flip-flop) and ends at a register (or flip-flop) within the block. The second netlist includes nets which are coupled to a pin of the block. Generally, the first netlist is ¾ of the initial block-level netlist while the second netlist is ¼ of the initial block-level netlist. If the second netlist ratio is greater than ¼, this indicates inefficient partitioning of the blocks.

Once the first netlist and the second netlist are obtain, an extraction operation to obtain parasitic resistance and capacitance is performed on the second netlist of each block. In an embodiment, the partitioning of the block-level netlist and the extraction operation in each block are performed in parallel. Moreover, an extraction operation is performed on the top-level netlist. In an embodiment, a software tool replaces the abutted pins of the top-level netlist with zero ohm resistors.

Some software tools utilized to perform the timing analysis are unable to operate on netlists having nets that are coupled to multiple pins of a block. In an embodiment of the present invention, these netlist are transformed by using "assign statements" to assign different names to the nets that are coupled to multiple pins of a block. Hence, each different named net can be coupled to a separate pin of the block.

In an embodiment, the second netlist and its associated extraction file of each block and the top-level netlist and its associated extraction file are utilized by software tools to perform the timing analysis. This timing analysis can be performed significantly faster than the case where the block-level netlist is not partitioned into the first netlist and the second netlist. In an embodiment, the timing graph resulting from the timing analysis can be analyzed to extract timing constraints (relating to the delay that can be generated by a block) for each block. Hence, if a block is optimized to meet its extracted timing constrains, the block is more likely to meet its timing parameter when the block interacts with the other blocks in the integrated circuit.

The foregoing descriptions of specific embodiments of the present invention have been presented for purposes of illustration and description. They are not intended to be exhaustive or to limit the invention to the precise forms disclosed, and obviously many modifications and variations are possible in light of the above teaching. The embodiments were chosen and described in order to best explain the principles of the invention and its practical application, to thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the Claims appended hereto and their equivalents.

What is claimed is:

1. A method of improving a physical design of a current integrated circuit, comprising the steps of:

a) receiving a netlist of said current integrated circuit;

b) receiving physical design information from a prior integrated circuit; and

c) generating said physical design based on said netlist and said physical design information.

2. A method as recited in claim **1** wherein said physical design information includes pin assignments of blocks of said prior integrated circuit.

3. A method as recited in claim **1** wherein said physical design information includes optimal clock distribution tree of said prior integrated circuit.

4. A method as recited in claim **1** wherein said physical design information includes parasitic extraction data of said prior integrated circuit.

5. A method as recited in claim **1** wherein said physical design information includes identification of congested blocks of said prior integrated circuit.

6. A method as recited in claim **1** wherein said physical design information includes metal resources of said prior integrated circuit.

7. A method as recited in claim **1** wherein said physical design information includes information which facilitates optimizing said current integrated circuit.

8. A method as recited in claim **1** wherein said step c) includes:

generating a top-level physical design of said current integrated circuit using said netlist and said physical design information including block-level physical design information of said prior integrated circuit.

9. A method as recited in claim **1** wherein said step c) includes:

generating a block-level physical design of said current integrated circuit using said netlist and said physical design information including block-level physical design information of said prior integrated circuit.

10. A method as recited in claim 1 wherein said physical design is an abutted-pin hierarchical physical design including a top-level physical design and a block-level physical design.

11. A method as recited in claim 1 wherein said physical design information includes locations of obstructions of said prior integrated circuit.

12. A method as recited in claim 11 wherein said obstructions include a random access memory (RAM).

13. A method as recited in claim 1 wherein said step c) includes:

partitioning said netlist into a plurality of blocks, each block including a block-level netlist;

performing a top-level floor planning;

performing a top-level placement and route for a plurality of top-level objects;

performing a top-level placement and route for a plurality of ports from said blocks to determine pin assignments for each block; and

generating and optimizing a block-level physical design for each block in parallel.

14. A method as recited in claim 13 wherein said generating and optimizing includes:

pressing each portion of each top-level object, which is located within a boundary of a particular block, into said particular block;

generating each pin for each block based on said top-level placement and route to determine pin assignments;

performing a block-level floor planning for each block;

performing a block-level placement for each block;

performing a plurality of block-level operations to optimize each block; and

performing a block-level route for each block.

15. A computer-readable medium comprising computer-executable instructions stored therein for performing a method of improving a physical design of a current integrated circuit, said method comprising:

a) receiving a netlist of said current integrated circuit;

b) receiving physical design information from a prior integrated circuit; and;

c) generating said physical design based on said netlist and said physical design information.

16. A computer-readable medium as recited in claim 15 wherein said physical design information includes pin assignments of blocks of said prior integrated circuit.

17. A computer-readable medium as recited in claim 15 wherein said physical design information includes optimal clock distribution tree of said prior integrated circuit.

18. A computer-readable medium as recited in claim 15 wherein said physical design information includes parasitic extraction data of said prior integrated circuit.

19. A computer-readable medium as recited in claim 15 wherein said physical design information includes identification of congested blocks of said prior integrated circuit.

20. A computer-readable medium as recited in claim 15 wherein said physical design information includes metal resources of said prior integrated circuit.

21. A computer-readable medium as recited in claim 15 wherein said physical design information includes information which facilitates optimizing said current integrated circuit.

22. A computer-readable medium as recited in claim 15 wherein said step c) includes:

generating a top-level physical design of said current integrated circuit using said netlist and said physical design information including block-level physical design information of said prior integrated circuit.

23. A computer-readable medium as recited in claim 15 wherein said step c) includes:

generating a block-level physical design of said current integrated circuit using said netlist and said physical design information including block-level physical design information of said prior integrated circuit.

24. A computer-readable medium as recited in claim 15 wherein said physical design is an abutted-pin hierarchical physical design including a top-level physical design and a block-level physical design.

25. A computer-readable medium as recited in claim 15 wherein said physical design information includes locations of obstructions of said prior integrated circuit.

26. A computer-readable medium as recited in claim 25 wherein said obstructions include a random access memory (RAM).

27. A computer-readable medium as recited in claim 15 wherein said step c) includes:

partitioning said netlist into a plurality of blocks, each block including a block-level netlist;

performing a top-level floor planning;

performing a top-level placement and route for a plurality of top-level objects;

performing a top-level placement and route for a plurality of ports from said blocks to determine pin assignments for each block; and

generating and optimizing a block-level physical design for each block in parallel.

28. A computer-readable medium as recited in claim 27 wherein said generating and optimizing includes:

pressing each portion of each top-level object, which is located within a boundary of a particular block, into said particular block;

generating each pin for each block based on said top-level placement and route to determine pin assignments;

performing a block-level floor planning for each block;

performing a block-level placement for each block;

performing a plurality of block-level operations to optimize each block; and

performing a block-level route for each block.

29. A method of determining a plurality of pins for each block of a physical design of a current integrated circuit, comprising:

a) receiving a netlist of said current integrated circuit;

b) receiving physical design information from a prior integrated circuit, wherein said physical design information includes pin assignments of blocks of said prior integrated circuit;

c) using said netlist and said physical design information to perform a top-level placement for a plurality of ports corresponding to each block of said current integrated circuit;

d) using said netlist and said physical design information to perform a top-level route for said ports to determine pin assignments for each block of said current integrated circuit; and

e) generating each pin for each block based on said top-level route to determine pin assignments.

30. A method as recited in claim **29** further comprising:

partitioning said netlist into a plurality of blocks of said current integrated circuit, each block including a block-level netlist.

31. A method as recited in claim **29** wherein said physical design information includes optimal clock distribution tree of said prior integrated circuit.

32. A method as recited in claim **29** wherein said physical design information includes parasitic extraction data of said prior integrated circuit.

33. A method as recited in claim **29** wherein said physical design information includes identification of congested blocks of said prior integrated circuit.

34. A method as recited in claim **29** wherein said physical design information includes metal resources of said prior integrated circuit.

35. A method as recited in claim **29** wherein said physical design information includes information which facilitates optimizing said current integrated circuit.

36. A method as recited in claim **29** wherein said physical design information includes locations of obstructions of said prior integrated circuit.

37. A method as recited in claim **36** wherein said obstructions include a random access memory (RAM).

38. A method as recited in claim **29** wherein said physical design is an abutted-pin hierarchical physical design.

39. A method as recited in claim **38** wherein said physical design includes a top-level physical design.

40. A method as recited in claim **38** wherein said physical design includes a block-level physical design.

41. A computer-readable medium comprising computer-executable instructions stored therein for performing a method of determining a plurality of pins for each block of a physical design of a current integrated circuit, comprising:

a) receiving a netlist of said current integrated circuit;

b) receiving physical design information from a prior integrated circuit, wherein said physical design information includes pin assignments of blocks of said prior integrated circuit;

c) using said netlist and said physical design information to perform a top-level placement for a plurality of ports corresponding to each block of said current integrated circuit;

d) using said netlist and said physical design information to perform a top-level route for said ports to determine pin assignments for each block of said current integrated circuit; and

e) generating each pin for each block based on said top-level route to determine pin assignments.

42. A computer-readable medium as recited in claim **41** wherein said method further comprises:

partitioning said netlist into a plurality of blocks of said current integrated circuit, each block including a block-level netlist.

43. A computer-readable medium as recited in claim **41** wherein said physical design information includes optimal clock distribution tree of said prior integrated circuit.

44. A computer-readable medium as recited in claim **41** wherein said physical design information includes parasitic extraction data of said prior integrated circuit.

45. A computer-readable medium as recited in claim **41** wherein said physical design information includes identification of congested blocks of said prior integrated circuit.

46. A computer-readable medium as recited in claim **41** wherein said physical design information includes metal resources of said prior integrated circuit.

47. A computer-readable medium as recited in claim **41** wherein said physical design information includes information which facilitates optimizing said current integrated circuit.

48. A computer-readable medium as recited in claim **41** wherein said physical design information includes locations of obstructions of said prior integrated circuit.

49. A computer-readable medium as recited in claim **48** wherein said obstructions include a random access memory (RAM).

50. A computer-readable medium as recited in claim **41** wherein said physical design is an abutted-pin hierarchical physical design.

51. A computer-readable medium as recited in claim **50** wherein said physical design includes a top-level physical design.

52. A computer-readable medium as recited in claim **50** wherein said physical design includes a block-level physical design.

* * * * *

ATTACHMENT E

US005757658A

United States Patent [19]
Rodman et al.

[11] Patent Number: 5,757,658
[45] Date of Patent: May 26, 1998

[54] PROCEDURE AND SYSTEM FOR PLACEMENT OPTIMIZATION OF CELLS WITHIN CIRCUIT BLOCKS BY OPTIMIZING PLACEMENT OF INPUT/OUTPUT PORTS WITHIN AN INTEGRATED CIRCUIT DESIGN

[75] Inventors: **Paul K. Rodman; Marjorie S. Levine**, both of Palo Alto, Calif.

[73] Assignee: **Silicon Graphics, Inc.**, Mountain View, Calif.

[21] Appl. No.: **611,785**

[22] Filed: **Mar. 6, 1996**

[51] **Int. Cl.**^{6} **G06F 17/50**
[52] **U.S. Cl.** **364/491**; 364/490
[58] **Field of Search** 364/488, 489, 364/490, 491

[56] **References Cited**

U.S. PATENT DOCUMENTS

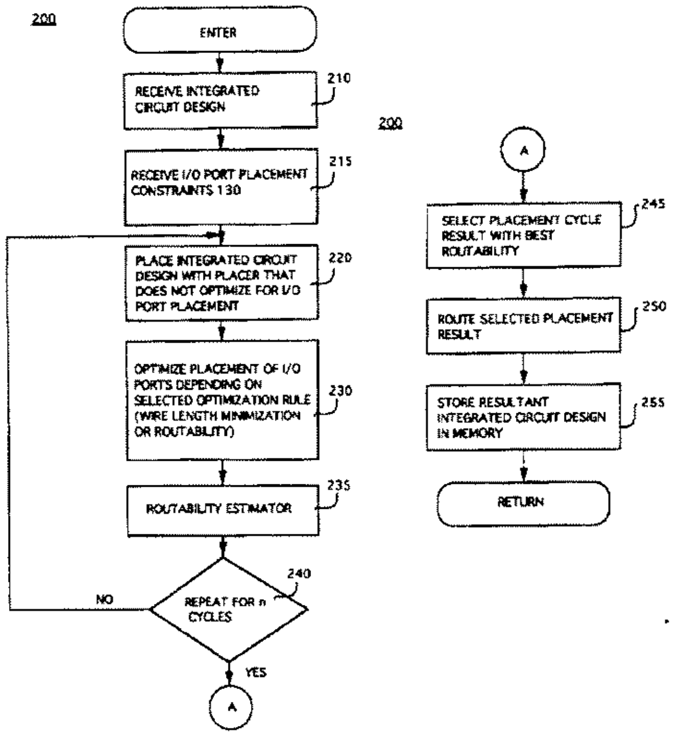
3,621,208	11/1971	Isett et al.	364/489
3,644,937	2/1972	Isett	364/489
3,702,004	10/1972	Eskew et al.	364/489
5,151,868	9/1992	Nishiyama et al.	364/490
5,544,088	8/1996	Aubertine et al.	364/489
5,548,525	8/1996	Albee et al.	364/489

Primary Examiner—Vincent N. Trans
Attorney, Agent, or Firm—Wagner, Murabito & Hao

[57] **ABSTRACT**

A system and procedure for placement optimization of input/output ports associated with edges of circuit blocks within an integrated circuit design. The integrated circuit design is composed of circuit blocks that communicate using inter-block signal wires coupled to input/output ports (IOPs) located along edges of circuit blocks. An arbitrary IOP placement is first received, e.g., from a global floorplanner, and indicates (1) the allowable edge placement domains for each IOP and can optionally include (2) an arbitrary IOP placement within these allowable edge domains. A cell placer (e.g., a quadratic based standard cell placer) receives the arbitrary IOP placement and, for each circuit block, places cells represented within internal netlists. The placer does not optimize the placement of the IOPs. For each IOP, the set of cells of the net that is coupled to the IOP is determined. Each IOP is then moved, within its allowable edge placement, to a position closest to the nearest cell that is within its associated net. The above sequence is then repeated a number of times (e.g., IOPs are moved and the placer is run again); upon each run the routability of the placement is estimated. After the above iterations, the present invention accepts the placement with the best estimated routability and this placement is then routed by a router. By taking into account the position of cells associated with an IOP, and displacing the IOP near these cells, the internal circuit is more efficiently placed which reduces the size of the circuit block up to 30 percent.

21 Claims, 9 Drawing Sheets


200
ENTER
RECEIVE INTEGRATED CIRCUIT DESIGN
210
RECEIVE I/O PORT PLACEMENT CONSTRAINTS 130
215
PLACE INTEGRATED CIRCUIT DESIGN WITH PLACER THAT DOES NOT OPTIMIZE FOR I/O PORT PLACEMENT
220
OPTIMIZE PLACEMENT OF I/O PORTS DEPENDING ON SELECTED OPTIMIZATION RULE (WIRE LENGTH MINIMIZATION OR ROUTABILITY)
230
ROUTABILITY ESTIMATOR
235
REPEAT FOR n CYCLES
240
NO
YES
A
200
A
SELECT PLACEMENT CYCLE RESULT WITH BEST ROUTABILITY
245
ROUTE SELECTED PLACEMENT RESULT
250
STORE RESULTANT INTEGRATED CIRCUIT DESIGN IN MEMORY
255
RETURN

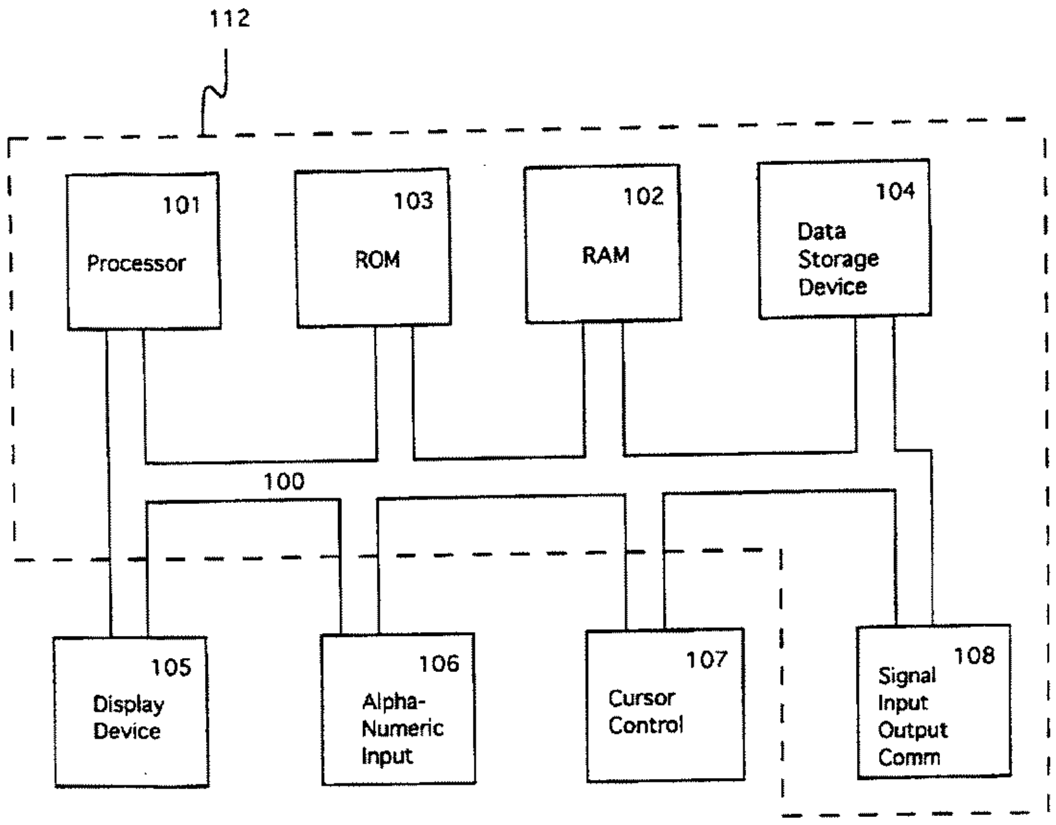

112
101
Processor
103
ROM
102
RAM
104
Data Storage Device
100
105
Display Device
106
Alpha-Numeric Input
107
Cursor Control
108
Signal Input Output Comm

FIG. 1

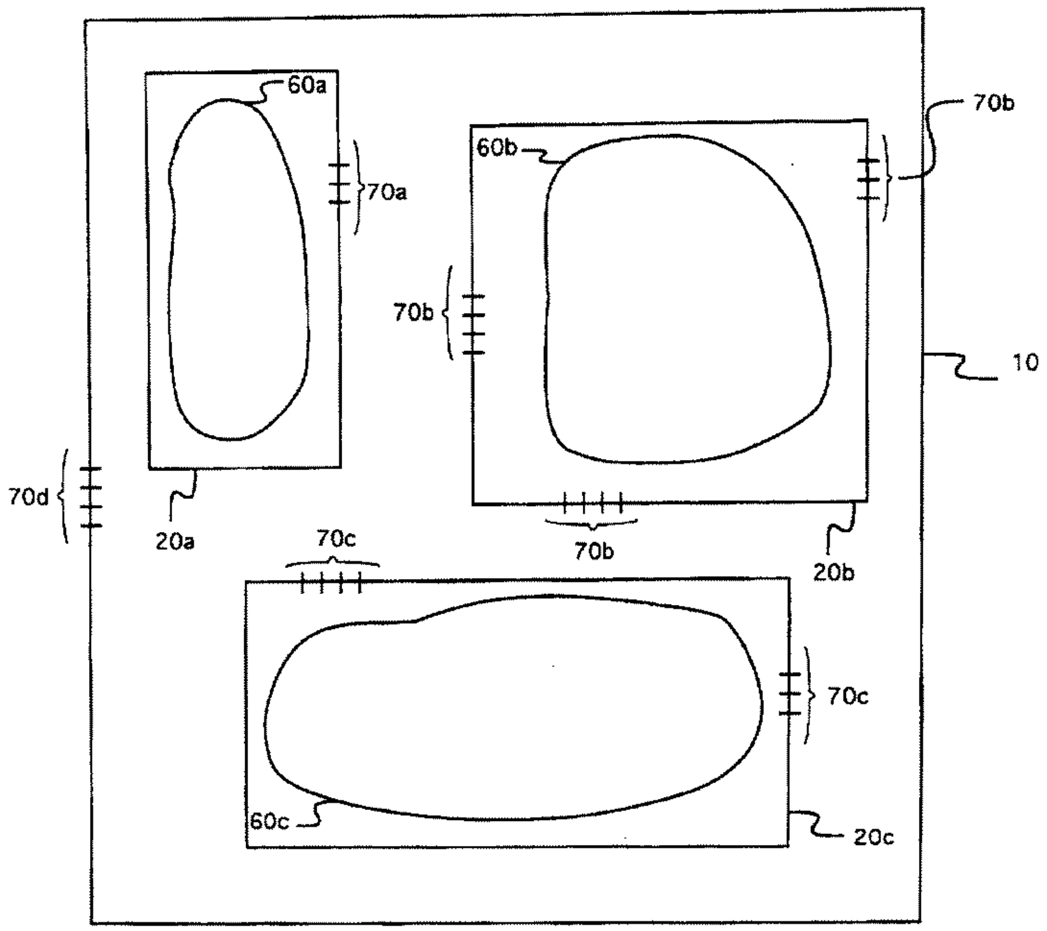


FIG. 2

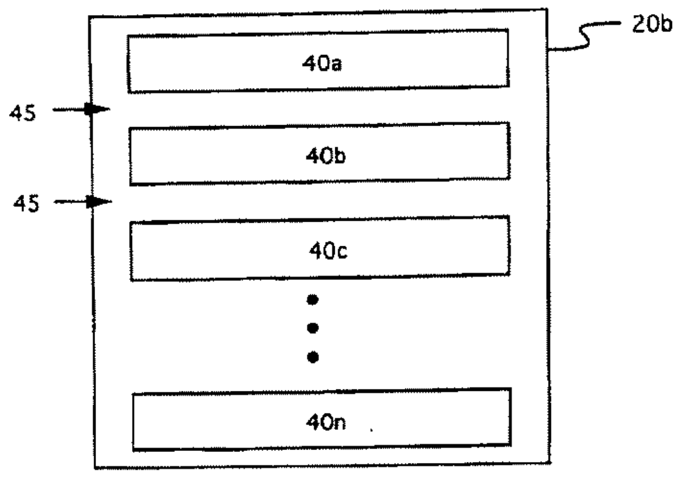

20b
40a
45
40b
45
40c
40n

FIG. 3A

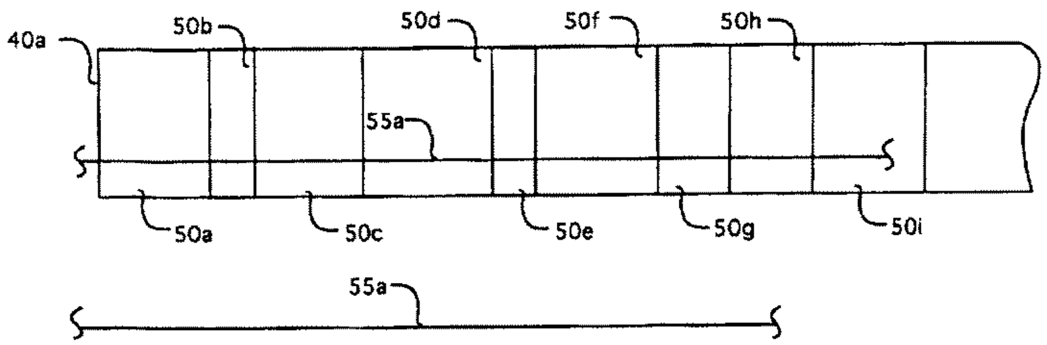

40a
50b
50d
50f
50h
55a
50a
50c
50e
50g
50i
55a

FIG. 3B

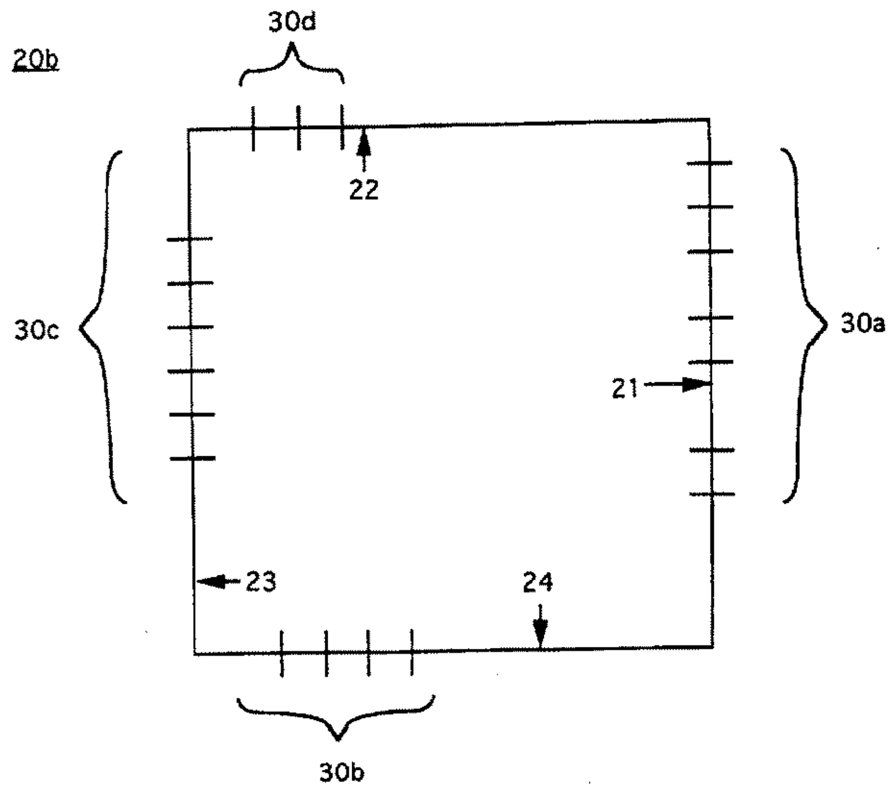


FIG. 4

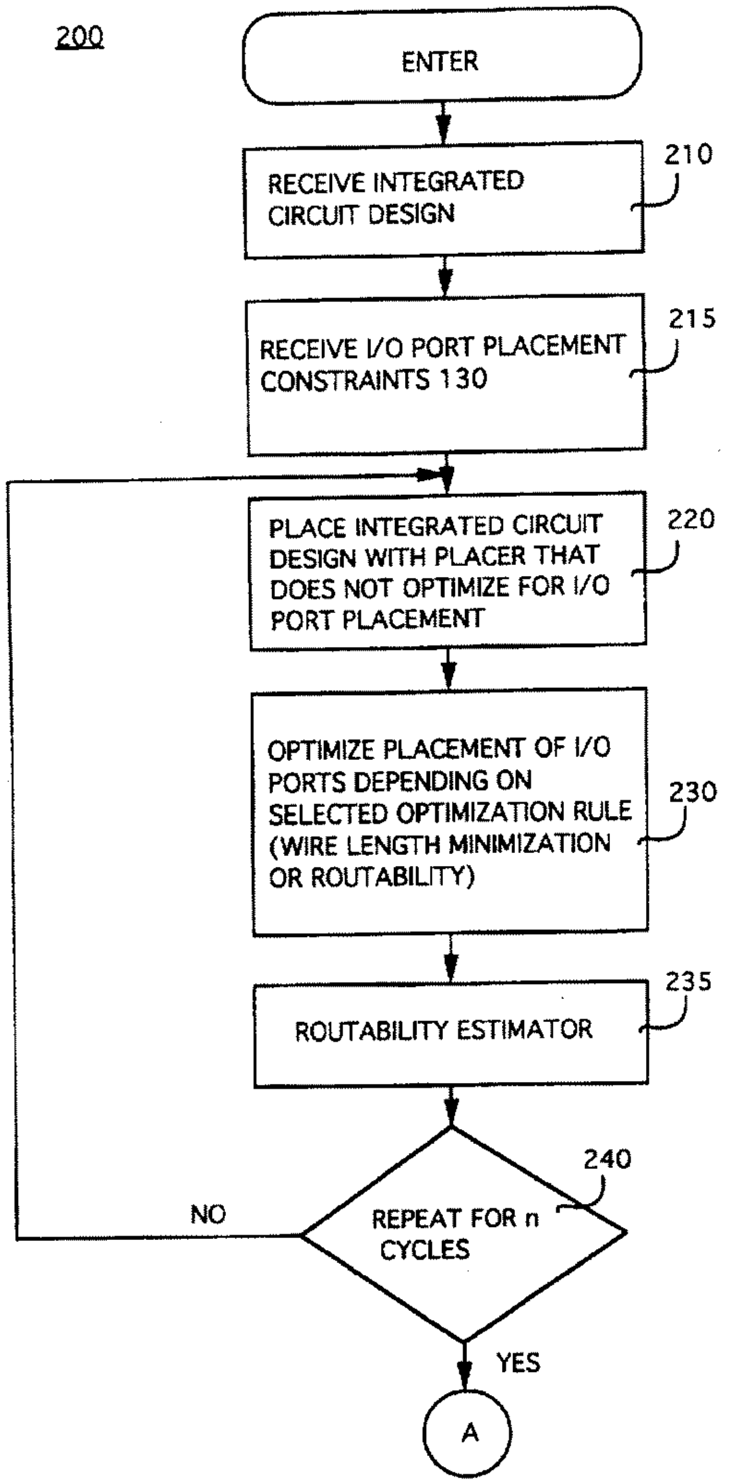

200
ENTER
RECEIVE INTEGRATED CIRCUIT DESIGN
210
RECEIVE I/O PORT PLACEMENT CONSTRAINTS 130
215
PLACE INTEGRATED CIRCUIT DESIGN WITH PLACER THAT DOES NOT OPTIMIZE FOR I/O PORT PLACEMENT
220
OPTIMIZE PLACEMENT OF I/O PORTS DEPENDING ON SELECTED OPTIMIZATION RULE (WIRE LENGTH MINIMIZATION OR ROUTABILITY)
230
ROUTABILITY ESTIMATOR
235
REPEAT FOR n CYCLES
240
NO
YES
A

FIG. 5A

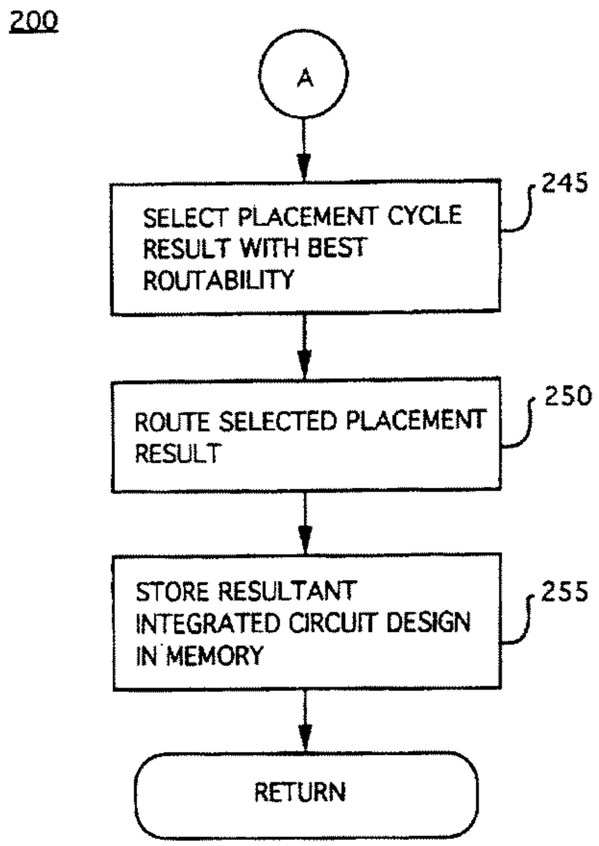

200
A
SELECT PLACEMENT CYCLE RESULT WITH BEST ROUTABILITY
245
ROUTE SELECTED PLACEMENT RESULT
250
STORE RESULTANT INTEGRATED CIRCUIT DESIGN IN MEMORY
255
RETURN

FIG. 5B

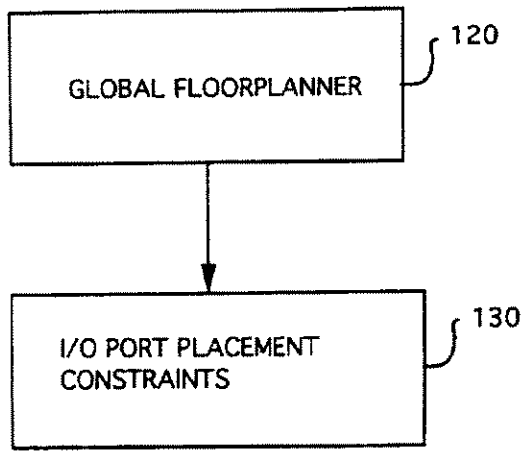

GLOBAL FLOORPLANNER
120
I/O PORT PLACEMENT CONSTRAINTS
130

FIG. 6

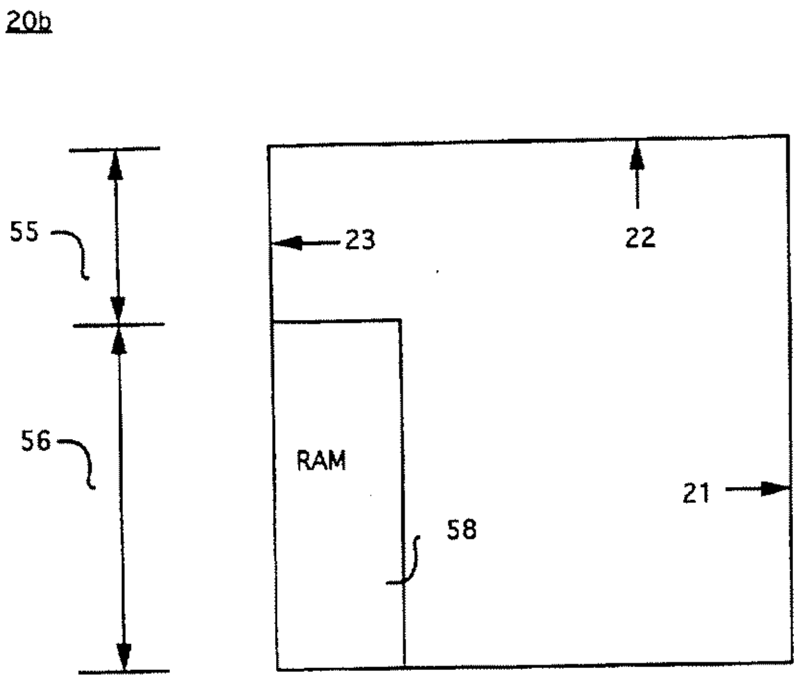


FIG. 7

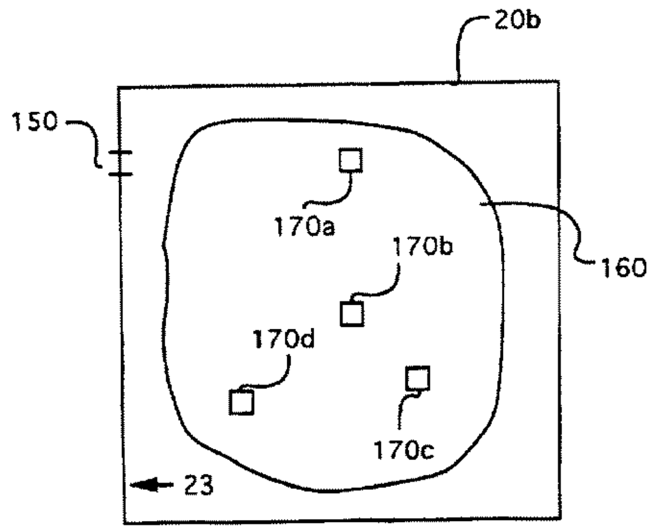

20b
150
170a
170b
160
170d
170c
23

FIG. 8A

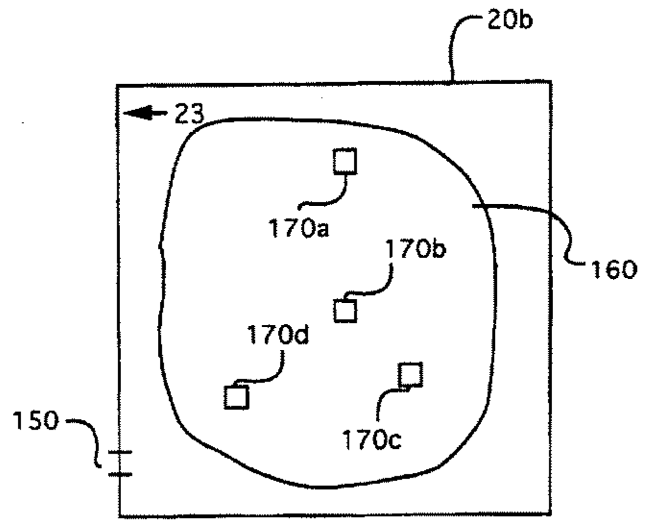

20b
23
170a
170b
160
170d
150
170c

FIG. 8B

PROCEDURE AND SYSTEM FOR PLACEMENT OPTIMIZATION OF CELLS WITHIN CIRCUIT BLOCKS BY OPTIMIZING PLACEMENT OF INPUT/OUTPUT PORTS WITHIN AN INTEGRATED CIRCUIT DESIGN

BACKGROUND OF THE INVENTION

1. Field Of The Invention

The present invention relates to the field of computer aided design tools used for designing integrated circuits. Specifically, the present invention relates to input/output port placement optimizations in conjunction with a computer implemented placer.

2. Related Art

Integrated circuits are designed using computer aided design (CAD) tools. The integrated circuit design process includes constructing the integrated circuit design out of simple circuits (e.g., "standard cells") that are connected together electrically using wire interconnects. The standard cells and connections between them are stored in well known databases called "netlists."

As part of the design process, the design information within a netlist is placed and routed by the CAD tool. The CAD tool utilizes placing and routing processes (also called placers and routers) that are typically software programs executed on the CAD tool. The placer determines the optimum location of each standard cell within the integrated circuit layout on the semiconductor surface. The placement location is optimized to reduce the distance between standard cells that are electrically connected to each other by wire interconnects (e.g., input/output lines). This is done to minimize semiconductor area consumed by the integrated circuit and is also done to minimize the lengths of wire interconnects to reduce net capacitance within the design. The router optimizes the routing of input/output lines between connected standard cells so that areas of the integrated circuit layout do not become overly congested by input/output lines.

An integrated circuit design is typically composed of several circuit blocks (FIG. 2) that are themselves composed of series of standard cells. The circuit blocks are connected to each other by nets. The nets are themselves coupled to input/output ports (IOPs) of the circuit blocks. According to the CAD tools, the IOPs are positioned along the edges of the circuit blocks.

Optimizing the placement of IOPs with respect to a circuit block is advantageous because the optimization reduces the size of the circuit block. A non-optimal placement of the IOPs can result in an excess of wire that adds area to the integrated circuit and increases product costs and may reduce clock rates (e.g., performance). However, optimizing the placement of the IOPs along the edge of a circuit block with respect to the internal circuitry of the circuit block posses a problem within CAD systems. Specifically, most CAD systems do not optimize the placement of IOPs with respect to the internals of the block at all. Other CAD systems that do optimize IOP placement require an extensive amount of processing time to complete.

One type of prior art placer is effective at placing the cells of the internal circuitry of an integrated circuit block, but this placer does not optimize the placement of the IOPs along the edges of the circuit block. Instead, this prior art placer is supplied a predefined IOP placement which is then not altered by the placer. The result is a circuit block that has a disadvantageous amount of routing area within its internal circuitry because the IOP placement was assigned without respect to the actual locations of the standard cells (the assignment was made before the internal circuitry was placed). The additional size is provided to accommodate extra wiring area used to properly connect the cells to their associated IOPs, which may be very distant from the cells. What is needed is a system that optimizes the placement of the IOPs with respect to the internal circuitry of a circuit block. The present invention provides this capability.

Another prior art placer (e.g., that performs simulated annealing) automatically optimizes the placement of the IOPs along the edges of the circuit blocks in an effort to minimize the length of the wire connections between the IOPs and the internal circuitry of the circuit block. In this prior art design, the locations of the IOPs can be adjusted along their respective edges during the placing process. However, the simulated annealing placer is not advantageous because it requires an extensive amount of processing time to complete and does not produce adequate placements for the internal circuitry due in part to this extended processing period. Also, the main objective of the simulated annealing placer is to minimize wire interconnect lengths which often creates signal routing problems within the overall design. What is needed is an efficient IOP placement optimization procedure that performs well for internal circuit block circuitry and that does not require an extensive amount of processing to complete and that does not create signal routing problems for the internal circuitry of the circuit block. The present invention provides this capability.

Accordingly, the present invention provides a novel system for effectively optimizing the placement of IOPs with respect to internal circuitry of a circuit block to reduce the size of the circuit block. The present invention provides a system that adjusts the IOP locations during the placement process. The present invention provides an effective IOP placement optimization procedure that does not require an extensive amount of processing time, yet optimizes circuit block netlists and IOP placements to effectively eliminate routing problems within the circuit blocks and also to effectively reduce the size of these circuit blocks. These and other advantageous of the present invention not specifically mentioned above will become clear within discussions of the present invention presented herein.

SUMMARY OF THE INVENTION

A computer implemented procedure and system are discussed for placement optimization of input/output ports associated with edges of circuit blocks within an integrated circuit design. Computer implemented placement and route procedures (e.g., "placers and routers") do not take into consideration the placement of input/output ports when optimizing the placement of cells within the integrated circuit design. The integrated circuit design is composed of several circuit blocks that communicate with each other using input/output lines coupled to input/output ports (IOPs) located along edges of circuit blocks.

Within the present invention, an arbitrary IOP placement is first received and can be generated from a global floor planner procedure (GFP) of a computer aided design tool. The arbitrary IOP placement indicates (1) the allowable edge placements for each IOP and (2) an arbitrary assignment of ports within their allowable edge placements. A computer implemented cell placer (e.g., a quadratic based standard cell placer) and computer implemented router receives the arbitrary IOP placement and, for each circuit block, places the cells which are represented within internal netlists and routes their interconnections. During placement and routing, the cells (e.g., standard cells) are placed so that the distance between connected cells is minimized without creating routing obstructions. The placer used in the present invention does not optimize the placement of the IOPs.

After the placer executes, for each IOP, the present invention identifies the net that is coupled to the IOP and the cells within this net. The integrated circuit design is then modified by moving each IOP within its allowable edge placement to a position closest to the nearest cell on the net that is coupled to the IOP. The above sequence is then repeated a number of times (e.g., IOPs are moved and the placer is run again); upon each run the routability of the placement is estimated. After the above iterations, the present invention accepts the placement with the best estimated routability and this placement is then routed by a router. By taking into account the position of cells that a given IOP is coupled to, and displacing each IOP near the closest cell, the present invention more efficiently places the internal circuitry of the circuit block reducing the size of unused area and wire interconnect area of the circuit block. In accordance with the present invention, circuit blocks can be reduced in size up to 30 percent.

Specifically, embodiments of the present invention include, a computer implemented method for placing cells within an integrated circuit design and comprising the steps of: (a) receiving a set of input/output port placement constraints including eligible edge placement domains individually associated with the input/output ports; (b) determining an arbitrary initial placement for each input/output port within its associated eligible edge placement domain provided the arbitrary placement is absent from the input/output port placement constraints; (c) placing the cells of each circuit block using a placer procedure that does not optimize input/output port placement; (d) optimizing placements of the input/output ports in accordance with an optimization rule by adjusting placements of the input/output ports within their eligible edge placement domains; (e) performing a routability estimate of the integrated circuit design based on results generated in step (d); (f) repeating steps (c)–(e) over n cycles; and (g) selecting a best routable optimized placement of the n cycles and routing the best routable optimized placement.

Embodiments of the present invention include the above and wherein the step (d) comprises the step of adjusting placements of each input/output port along its eligible edge placement domain to a placement position closest to a coupled cell that is nearest to the eligible edge placement domain of the each input/output port. Embodiments of the present invention include the above and wherein the step (d) comprises the further step of adjusting placement of each input/output port along its eligible edge placement domain to a placement position that relieves routing congestion. Embodiments also include a computer system implemented in accordance with the above.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram of a general purpose computer system that can be used within a computer aided design (CAD) tool for designing, placing and routing integrated circuits in accordance with the present invention.

FIG. 2 is a block diagram illustration of an integrated circuit layout including several exemplary circuit blocks used by the present invention.

FIG. 3A is a block diagram illustration of an integrated circuit layout of a circuit block including several rows of standard cells used by the present invention.

FIG. 3B illustrates an integrated circuit layout of an exemplary row of standard cells used by the present invention.

FIG. 4 is a block diagram illustrating input/output port assignments along edges of a circuit block in accordance with the present invention.

FIG. 5A is a first part of a flow diagram illustrating processing logic of one embodiment of the present invention.

FIG. 5B is a second part of a flow diagram illustrating processing logic of one embodiment of the present invention.

FIG. 6 is a flow diagramming illustrating an input/output port assignment data used by one embodiment of the present invention and generated by a global floorplanner.

FIG. 7 is a block diagram illustration eligible input/output port assignments along edges or partial edges of a circuit block in accordance with the present invention.

FIG. 8A illustrates an exemplary circuit block having a netlist that includes several cells that are coupled to an exemplary input/output port.

FIG. 8B illustrates an exemplary circuit block of FIG. 8A with the exemplary input/output block optimized in accordance with the present invention.

DETAILED DESCRIPTION OF THE INVENTION

In the following detailed description of the present invention, numerous specific details are set forth in order to provide a thorough understanding of the present invention. However, it will be obvious to one skilled in the art that the present invention may be practiced without these specific details or with equivalents thereof. In other instances, well known methods, procedures, components, and circuits have not been described in detail as not to unnecessarily obscure aspects of the present invention.

NOTATION AND NOMENCLATURE

Some portions of the detailed descriptions which follow are presented in terms of procedures, logic blocks, processing, steps, and other symbolic representations of operations on data bits within a computer memory. These descriptions and representations are the means used by those skilled in the data processing arts to most effectively convey the substance of their work to others skilled in the art. A procedure, logic block, process, etc., is generally conceived to be a self-consistent sequence of steps or instructions leading to a desired result. The steps require physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated in a computer system. It has proven convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers, or the like.

It should be borne in mind that all of the above and similar terms are to be associated with the appropriate physical quantities they represent and are merely convenient labels applied to these quantities. Unless specifically stated otherwise as apparent from the following discussions, it is appreciated that throughout the present invention, discussions utilizing terms such as "processing" or "computing" or "calculating" or "determining" or "displaying" or the like, refer to the action and processes of a computer system, or similar electronic computing device, that manipulates and transforms data represented as physical (electronic) quantities within the computer system's registers and memories into other data similarly represented as physical quantities within the computer system memories or registers or other such information storage, transmission or display devices.

CAD COMPUTER SYSTEM

The present invention includes a system and a computer implemented process, in conjunction with a placer, for effectively optimizing the placement of input/output ports (IOPs) located on the perimeter of a circuit block with respect to internal cells that are electrically coupled to the IOPs. The process of the present invention is implemented within a computer aided design (CAD) tool which includes a computer system such as exemplary computer system **112** shown in FIG. 1.

With reference to FIG. 1, the exemplary computer system **112** includes a bus **100** for communicating information, a central processor **101** coupled with the bus **100** for processing information and instructions and a volatile memory **102** (e.g., a random access memory) coupled with the bus **100** for storing information and instructions for the central processor **101**. Computer system **112** also includes a nonvolatile memory **103** (e.g., a read only memory **103**) which is coupled with the bus **100** for storing static information and instructions for the processor **101**, a data storage device **104** such as a magnetic or optical disk and disk drive coupled with the bus **100** for storing information and instructions, and a display device **105** coupled to the bus **100** for displaying information to the computer user.

The display device **105** of FIG. 1 utilized with the computer system **112** can be a liquid crystal device, cathode ray tube, or other display device suitable for creating graphic images and alphanumeric characters recognizable to the user. Computer system **112** also includes an alphanumeric input device **106** including alphanumeric and function keys coupled to the bus **100** for communicating information and command selections to the central processor **101**, a cursor control device **107** coupled to the bus for communicating user input information and command selections to the central processor **101**, and a signal generating device **108** (e.g., a modem) coupled to the bus **100** for receiving and sending information to and from the processor **101**.

CIRCUIT BLOCKS AND STANDARD CELLS

FIG. 2 illustrates an exemplary custom integrated circuit layout **10** including several custom circuit blocks **20**_a_–**20**_c_. Each circuit block includes IOPs **70**_a_–**70**_c_, located and modeled along the edges of the circuit block, as well as internal circuitry represented by netlists **60**_a_–**60**_c_. Typically, more than one circuit block are used to represent an entire integrated circuit. The IOPs **70**_a_–**70**_c_ couple to interblock signal wires (not shown in FIG. 2) that allow the circuit blocks to communicate with each other. IOPs **70**_d_ are also located along the layout **10** and allow signals to be communicated from and to circuits located off-chip. These IOPs **70**_d_ are typically referred to as primary input and primary outputs.

With reference to FIG. 3A, an exemplary circuit block **20**_b_ is shown in more detail. The internal netlist **60**b of custom circuit block **20**_b_ contains a number of rows **40**_a_–**40**_n_ of standard cell circuits. Each row **40**_a_–**40**_n_ can contain many standard cells. Routing channels **45** can be created between the rows **40**_a_–**40**_n_ and are used to allow flexible wire interconnect routing between the standard cells

of the rows. A typical IOP of a circuit block can be coupled to one or more standard cells. A group of standard cells coupled to the same IOP are said to be part of a single net.

FIG. 3B illustrates an exemplary row **40**_a_ in more detail. As shown, row **40**_a_ is composed of several standard cell circuits **50**_a_–**50**_i_. Each standard cell can have a variable width, as shown, but each standard cell shares the same height allowing uniform joining and packing. These geometry restrictions are employed to allow efficient placement of the standard cells during the placing process. As is well known in the art, each standard cell realizes a particular circuit function, e.g., flip-flop, AND gate, OR gate, XOR gate, multiplexer, etc., and the circuitry used to define a particular cell is stored in libraries within the computer system **112** of the CAD tool. Signal lines connecting standard cells can be placed inside the rows (e.g., line **55**_a_) and/or outside the rows (e.g., line **55**_b_).

During the placing process of the present invention, computer system **112** places the standard cells within rows in an attempt to minimize the distance between standard cells that are coupled together via input/output lines.

INPUT/OUTPUT PORTS OF A CIRCUIT BLOCK

With reference to FIG. 4, an exemplary circuit block **20**_b_ is shown with its IOPs indicated in more detail along the edges of circuit block **20**_b_. The placing process of the present invention models the IOPs as points along the edges of the circuit block **20**_b_. Each IOP has an associated eligible edge placement domain indicates the edge, edges or portion of an edge on which it is eligible to be located. Not every edge need have IOPs and, as shown, the IOP distribution in number and in location need not be uniform across different edges of a circuit block. In the example of FIG. 4, edge **21** contains IOPs **30**_a_, edge **24** contains IOPs **30**_b_, edge **23** contains IOPs **30**_c_, and edge **22** contains IOPs **30**_d_. Although not shown, each IOP connects to an input/output wire connect to allow signals to be interchanged between other circuit blocks of the integrated circuit layout **10** (FIG. 2).

PREFERRED CELL PLACEMENT PROCESS OF THE PRESENT INVENTION

FIG. 5A and FIG. 5B illustrate the placing process **200** of the present invention for optimizing IOP placements within an integrated circuit to provide reduced size circuit blocks. Process **200** is implemented within computer system **112** and includes several logic blocks which are executed by computer system **112**. Program (e.g., instruction) code for implementing the process **200** is stored in memory unit **102** and is executed by processor **101** (FIG. 1).

Process **200** starts at logic block **210** where process **200** causes the computer system **112** to receive an integrated circuit design. In one embodiment, this integrated circuit design is represented as netlist. The integrated circuit design is stored in a memory unit (e.g., memory **102** or **104**) of system **112**. The integrated circuit design (FIG. 2) includes several circuit blocks (e.g., more than one) which communicate to each other over inter-block signal wires. Each inter-block signal wire is coupled to IOPs located along the perimeter of circuit blocks. Generally, two or more IOPs are coupled to each inter-block signal wire because an inter-block signal wire can be connected to two or more circuit blocks. Each circuit block also contains a netlist description which defines its internal circuitry.

At logic block **215** of FIG. 5A, the present invention causes the computer system **112** to receive a set of predetermined IOP placement constraints **130** (FIG. 6) pertinent to the integrated circuit design received at block **210**. The IOP placement constraints **130** can originate from a computer memory unit or can be directly generated and supplied from a global floorplanner procedure (e.g., process) **120** as shown in FIG. **6**. The global floorplanner process **120**, in one embodiment, is also implemented as program (e.g., instruction) code that is executed on computer system **112** (FIG. 1).

The global floorplanner process **120** of FIG. **6** receives information indicating the positions of the circuit blocks (e.g., **20***a*, **20***b*, **20***c*) within the integrated circuit design. Knowledgeable of the positions of the circuits blocks, the global floor planner process **120** of FIG. **6** determines a number of IOP placement constraints **130** that are pertinent to the placement of the IOPs along the periphery of the circuit blocks. Within IOP placement constraints **130**, the global floor planner **120** generates an eligible placement domain for each IOP and optionally produces an arbitrary initial placement for the IOPs within their eligible edge domain, an IOP ordering constraint for the IOPs of each eligible edge domain and a maximum IOP density constraint allowed for each eligible edge domain. An exemplary data format of the IOP placement constraints **130** is described further below and illustrated in Table I.

With reference to FIG. **6**, any of a number of well known global floorplanner processes **120** can be used consistent with the scope of the present invention. Among its other functions, the global floorplanner process **120** places the circuit blocks **20***a*–**20***c* of the input integrated circuit layout **10** and is therefore aware of their locations with respect to each other.

The eligible edge domain for a given IOP indicates the possible edge locations (eligible edge placement domain) of a particular circuit block at which the IOP can be placed. Specifically, the eligible edge placement domain for each IOP indicates the possible locations along the edge, portion of an edge, or edges of a circuit block at which the IOP can be placed. The possible edge locations for a given IOP are determined based on the location of the circuit block or blocks with which the IOP communicates. For instance, if the particular IOP of a first circuit block is used to communicate with a second circuit block located to the right of the first circuit block, then the global floorplanner process **120** specifies that the particular IOP should be located on the right edge of the first circuit block. The right edge of the first circuit block is then the eligible edge placement domain for the particular IOP.

In addition to providing eligible edge domains for each IOP, the global floorplanner process **120** in one embodiment also specifies a particular edge location (e.g., initial edge placement) of the IOPs within their eligible edge domains. However, this initial placement is arbitrary and is not based on any particular preference. In other words, since internal circuitry of the circuit blocks (e.g., **20***a*, **20***b*, **20***c*) are not placed at the time the global floorplanner **120** is executed, any initial placement of the IOPs generated by the global floorplanner **120** is arbitrary. Although arbitrary, the above initial IOP placement can be viewed as an initial IOP placement constraint.

As described above, the global floor planner **120** can also provide other IOP constraints with respect to the placement of the IOPs within their individual eligible edge domains. One IOP constraint used is the ordering of IOPs within a particular eligible edge domain. A particular IOP ordering can be important to avoid signal wire routing problems (e.g., wires crossing, etc.). Another IOP constraint that can be

supplied by the global floorplanner is the maximum pin density allowed for any eligible edge domain. This information is important to maintain predetermined area constraints pertinent to the integrated circuit design layout **10**.

Table I below illustrates an exemplary format for the IOP constraint information stored in the IOP placement constraints **130** as supplied by the global floorplanner **120** and received by step **215** of the present invention process **200**.

TABLE I

IOP #	Eligible Edge Domain Definition	Arbitrary Placement	IOP Order	Max Density
IOP1	Edge domain for IOP1	(x, y) IOP1	IOP1 Order	Density1
IOP2	Edge domain for IOP2	(x, y) IOP2	IOP1 Order	Density2
IOP3	Edge domain for IOP3	(x, y) IOP3	IOP1 Order	Density3
IOPn	Edge domain for IOPn	(x, y) IOPn	IOP1 Order	Densityn

Where: IOPn Order is the pin order for IOPn within its Edge domain Densityn is the maximum pin density allowed for the Edge domain for IOPn (x, y) IOPn is the arbitrary placement for IOPn within its Edge domain

FIG. **7** illustrates that an eligible edge placement domain for a particular IOP can consist of: (1) a particular edge, e.g., edge **22**; or (2) more than one edge, e.g., edges **21** and **22**; or (3) only a portion of an edge, e.g., portion **22** of edge **23**. For instance, if a particular IOP for circuit block **20***b* receives a global signal, e.g., a clock which is easily obtained from a number of different sources, the global floor planner **120** indicates that this IOP can be located on edges **22** or **21** (or perhaps all edges of the circuit block). In this case, the eligible edge placement domain for this IOP contains multiple edges, e.g., edges **22** and **21**. In a second example, assume another IOP of circuit block **20***b* is to be coupled to a second circuit block located to the left of circuit block **20***b*. However, a RAM unit **58** is located on the left edge **23** and obstructs region **56** of left edge **23**. This exemplary IOP cannot be located in region **56** of edge **23**. Therefore, the eligible edge placement domain for this IOP is only region **55** of edge **23** which is an edge portion. In a third example, assume another IOP of circuit block **20***b* is to be coupled to a third circuit block located above circuit block **20***b*. In this case, the eligible edge placement domain for this IOP is edge **22**. In each of the above examples, the global floor planner **120** indicates the appropriate eligible edge placement domain for the particular IOPs discussed above.

With reference to the present invention process **200** of FIG. 5A, at logic block **215**, the IOP placement constraints **130** are received from the global floorplanner **120** and, as discussed above, indicate (1) the eligible edge domains for each IOP of the integrated circuit design and can also indicate: (2) an arbitrary IOP placement within each eligible edge domain; (3) an IOP ordering constraint for each eligible edge domain; and (4) a maximum IOP density constraint allowed for each eligible edge domain.

If not already performed by the global floorplanner process **120**, also at block **215**, the present invention determines an arbitrary initial placement of the IOPs with respect to their eligible edge domains. This initial IOP placement takes into consideration any other IOP constraints supplied by the global floorplanner **120** in the IOP placement constraints **130** (e.g., IOP ordering and maximum IOP density). At block **215**, any arbitrary or random placement can be used by the present invention and, for each IOP, an actual location is determined within its eligible edge placement domain. In one embodiment, the IOP reference name is used as the basis for the arbitrary first placement and the IOPs are equally spaced apart within their eligible edge placement domain. In another embodiment of the present invention, a random or pseudo random procedure can be used to perform the arbitrary placement of IOPs at block 215.

At logic block 220 of FIG. 5A, the standard cells of each circuit block are placed by the present invention. It is appreciated that the placer procedure used in block 220 does not optimize the placements of the IOPs that it receives. Although the placement procedure 220 does not optimize placement of the IOPs, the positions of the IOPs have an effect on the placement of the standard cells within the internal circuitry of each circuit block. This is the case because the wire connection length between a given IOP and its connected cells is one of the lengths that is minimized according to the placing process 220.

In accordance with the present invention, a number of well known placement procedures can be used at block 220 including a number of different quadratic based placement procedures. One such placement system is called q_place and is supplied by Cadence of California as part of its Cell3 tools. Another placer that is suited for use within the present invention is described in U.S. Pat. No. 5,267,176 by Antreich et al., issued on Nov. 30, 1993 and incorporated herein. Another placement procedure that can be used in accordance with the present invention at step 220 is described in a reference entitled "GORDIAN: VLSI Placement by Quadratic Programming and Slicing Optimization," by J. M. Kleinhans, G. Sigl, F. M. Johannes, and K. J. Antreich, published in IEEE Transactions on Computer Aided Design, Volume 10, No. 3, Mar. 1991. Other well known placement procedures that can be used in accordance with the present invention are described in a reference entitled "A procedure for Placement of Standard-Cell VLSI Circuits," by A. E. Dunlop and B. W. Kernighan, published in IEEE Transactions on Computer Aided Design, Volume CAD-4, No. 1, Jan. 1985 and in another reference entitled "PROUD: A fast Sea-of-Gates Placement Algorithm" by R. S. Tsay, E. S. Kuh, and C. P. Hsu, published by the 25th ACM/IEEE Design Automation Conference (Paper 22.3, 1988 IEEE).

At the completion of block 220 of FIG. 5A, each standard cell within the internal circuitry of each circuit block is given an initial placement within each circuit block (e.g., 20a, 20b, 20c). This initial placement is stored in a memory unit (e.g., unit 102 or 104) of computer system 112.

At logic block 230, the present invention displaces the positions of the IOPs of the integrated circuit design in order to optimize their positions according to a user-selected optimization rule. At block 230, in accordance with one user selected optimization rule, the positions of the IOPs are displaced to minimize wire length. Under the wire length minimization optimization, the present invention adjusts the locations of the IOPs to bring them closer to the cells to which they are connected to optimize their placement. At block 230, for each IOP of each circuit block, the present invention first determines the set of standard cells that are connected to the net that is connected to the IOP; this is performed only during the first pass of block 220 as multiple passes do not alter the composition of these nets. This cell association is stored in computer memory. FIG. 8A illustrates an example circuit block 20b with an exemplary IOP 150. As shown, there are four standard cells 170a–170d identified within the net coupled to IOP 150 within internal circuitry 160 of the circuit block 20b. The above cell identification is performed for each IOP of the integrated circuit design.

In accordance with wire length minimization optimization, once the set of standard cells is identified for

each IOP of each circuit block, the present invention determines, for each IOP, which cell of the set of standard cells associated with an IOP is closest to the eligible edge placement domain for the IOP. Once this closet cell is identified, the present invention adjusts the position of the IOP, along the eligible edge placement domain, until the IOP is closest to this selected cell. This effectively moves the IOP along its eligible edge placement domain to a position closest to its nearest coupled cell. Upon the first iteration of block 230, the present invention insures that extra wiring metal is not used in the circuit blocks by optimizing the IOP placements.

FIG. 8B illustrates an exemplary IOP displacement in accordance with step 230 for the user-selected wire length minimization optimization rule. Assume the eligible edge for IOP 150 is left edge 23. The cell closest to the left edge 23 is cell 170d. This cell is identified in block 230. The present invention moves the position of IOP 150 along edge 23 to a position nearest cell 170d as shown in FIG. 8B. The above IOP displacement is performed by the present invention for each IOP for each circuit block of the input integrated circuit design. At the completion of block 230, the present invention effectively places the IOPs closer to the standard cells to which they communicate. This reduces the wire interconnect lengths between the IOPs and their associated nets.

At block 230 of FIG. 5A, if two or more IOPs are placed at the same location along an edge of a particular circuit block, the present invention slightly displaces the overlapping IOPs until they no longer conflict.

Also at block 230, the user can select to perform IOP placement optimization in accordance with an optimization rule that optimizes for routability in conjunction with wire length minimization or instead of wire length minimization. In this case, the present invention is allowed to displace an IOP to a position (e.g., within its eligible edge domain) that does not minimize its associated wire length, but rather reduces wire congestion problem in identified areas. By reducing the wire conjunction, routability is improved.

It is appreciated that under any user-selected optimization rule for logic block 230, the present invention respects any IOP placement constraints 130 given by the global floorplanner process 120. At the completion of each iteration of block 230, the present invention records the IOP placement in computer memory within computer system 112.

At the completion of the IOP placement optimization 230 of FIG. 5A, logic block 235 of the present invention causes the computer system 1 12 to perform a routability estimate on the placement performed at block 220, as optimized by block 230, for the current iteration or cycle. It is appreciated that a number of well known routing estimator procedures can be employed within the present invention at block 235 of FIG. 5A to produce the routability estimate for the current cycle. The results of the routability estimate for each cycle are recorded in a memory unit of computer system 112 for subsequent reference.

In accordance with a preferred embodiment of the present invention, block 240 causes blocks 220, 230 and 235 to be executed for a predetermined number of cycles, n. According to wire length minimization optimization, upon each iteration of block 220, the standard cells within a particular circuit block gradually are placed (e.g., moved) to a location where the standard cell would naturally have been placed but for the random or arbitrary initial placement of the IOPs at block 215. The routability estimator 235 records its estimates of the routability for each cycle. The IOP placement optimization block 230 also records its IOP placement for each cycle. After n cycles have been executed, processing flows to block 245 of FIG. 5B.

At logic block 245 of FIG. 5B, the present invention process 200 causes computer system 112 to examine the results stored by the routability estimator 235 to determine which cycle of the n cycles produced the best routability estimate. At block 245, the IOP placement (as recorded by IOP placement optimization block 230) is selected having the best routability estimate over the n cycles. This selected IOP placement is forwarded to step 250.

At logic block 250, the present invention then performs a routing step of the IOP placement that is selected in block 245. At the completion of block 220, wire connections are routed between standard cells and IOPs and between circuit blocks and primary inputs and outputs. It is appreciated that a number of well known routing procedures can be employed within the present invention at block 250 of FIG. 5B. Once such router that can be employed at block 250 is called f_route and is supplied by Cadence of California.

At the completion of step 250, logic block 255 is executed wherein the present invention stores the placed and routed design of the integrated circuit design into a memory of the computer system 112.

ALTERNATIVE EMBODIMENT OF PROCESS 200

In accordance with an alternative embodiment of the present invention, circuit stabilization is utilized to determine the value of n as used by block 240 of FIG. 5A. Under this alternative embodiment, assuming a second iteration of block 220 and 230 has been performed, at logic block 240, the present invention determines if the standard cells of the input integrated circuit design were displaced in excess of a predetermined degree between the start and the end of block 220. The predetermined degree can be user determined. At block 240 of FIG. 5A, if the standard cells were displaced in excess of the predetermined degree, then the integrated circuit design is not yet stable and processing flows to logic block 220 of FIG. 5A where another placement is performed and another IOP displacement is performed at block 230. In typical practice, the integrated circuit can substantially stabilize after 4 or 5 iterations through logic block 220 and logic block 230. Upon each iteration of block 220, the standard cells within a particular circuit block gradually are placed (e.g., moved) to a location where the standard cell would naturally have been placed but for the random or arbitrary initial placement of the IOPs at block 215.

At block 240 of FIG. 5A, assuming a second iteration of block 220 and 230 has been performed, if the standard cells of the integrated circuit design were not displaced in excess of the predetermined degree, then the integrated circuit design is said to be stable and processing flows to logic block 250 of FIG. 5B. At logic block 250 of FIG. 5B, the present invention routes the stabilized design (see above) and at block 255 the present invention stores the resultant integrated circuit design in a memory unit (e.g., 102 or 104) of computer system 112. Processing then returns and exits placement procedure 200.

It is appreciated that another embodiment of the present invention is realized wherein block 220 and block 230 are executed a predetermined number of times (e.g., executed twice) without regard for the stability of the standard cells within the integrated circuit. After, block 250 of FIG. 5B is entered.

IOP OPTIMIZATION IN CONJUNCTION WITH A PLACER THAT DOES NOT PERFORM IOP OPTIMIZATION

By application of the post placement-processing step 230, where IOP are displaced along their eligible edge to the

nearest cell to which they couple, the present invention provides a more effective standard cell placement procedure used in conjunction with a placer process that does not handle IOP optimization. Upon subsequent placement iterations, the standard cells are placed in positions that utilize less semiconductor area because wasted wire connect area is reduced due to IOP placement optimization. In practice, the present invention allows designs to be reduced from 10 to 30 percent in area over prior art designs that do not provide placement optimization of IOPs, depending on the number of IOPs versus other nets and the type of netlist.

The preferred embodiment of the present invention, an effective IOP placement procedure is described using an existing placer to generate a first standard cell placement and then moving the location of the IOPs within their allowable edge placements such that each IOP is moved to a position closest to the nearest cell coupled to the IOP, and then subsequently executing the placer again. While the present invention has been described in particular embodiments, it should be appreciated that the present invention should not be construed as limited by such embodiments, but rather construed according to the below claims.

What is claimed is:

1. A computer implemented method of placing cells within an integrated circuit design, said integrated circuit design having circuit blocks including cells and input/output ports, said method comprising the steps of:

(a) receiving a set of input/output port placement constraints including eligible edge placement domains individually associated with said input/output ports;

(b) determining an arbitrary initial placement for each input/output port within its associated eligible edge placement domain provided said arbitrary placement is absent from said input/output port placement constraints;

(c) placing said cells of each circuit block using a placer procedure that does not optimize input/output port placement;

(d) optimizing placements of said input/output ports in accordance with an optimization rule by adjusting placements of said input/output ports within their eligible edge placement domains;

(e) performing a routability estimate of said integrated circuit design;

(f) repeating steps (c)–(e) over n cycles; and

(g) selecting a best routable optimized placement of said n cycles and routing said best routable optimized placement.

2. A method as described in claim 1 wherein said step (d) comprises the step of adjusting placements of each input/output port along its eligible edge placement domain to a placement position closest to a coupled cell that is nearest to said eligible edge placement domain of said each input/output port.

3. A method as described in claim 2 wherein said step of adjusting placements of each input/output port along its eligible edge placement domain comprises the steps of:

identifying a set of cells within a net coupled to said input/output port;

determining which cell of said set of cells is closest to said eligible edge placement domain of said input/output port, said cell being said coupled cell; and

adjusting a placement of said input/output port to said placement position closest to said coupled cell.

4. A method as described in claim 2 wherein said step (d) comprises the further step of adjusting placement of each

input/output port along its eligible edge placement domain to a position that reduces routing congestion within each circuit block.

5. A method as described in claim 1 wherein said placer procedure of step (c) is a quadratic based placement procedure minimizing wire connections between coupled cells of each circuit block and between input/output ports coupled to cells.

6. A method as described in claim 1 wherein said cells of said integrated circuit design are standard cells of variable width and uniform height.

7. A method as described in claim 1 wherein said set of eligible edge placement domains associated with said input/output ports are generated by a global floorplanner procedure.

8. A method as described in claim 1 wherein said input/output port placement constraints include input/output port ordering constraints.

9. In a computer system having a processor coupled to a bus and a memory coupled to said bus, a computer implemented method of placing cells within an integrated circuit design, said method comprising the steps of:

(a) receiving said integrated circuit design comprising circuit blocks having internal cells and edge located input/output ports;

(b) receiving input/output placement constraints including a set of eligible edge placement domains individually associated with said input/output ports;

(c) placing said cells of each circuit block using a placer procedure that does not optimize placement positions of said input/output ports;

(d) optimizing placements of said input/output ports to reduce wire length and reduce routing congestion by adjusting placements of said input/output ports within their eligible edge placement domains;

(e) performing a routability estimate of said integrated circuit design;

(f) repeating steps (c)-(e) for n cycles; and

(g) selecting a best routable optimized placement of said n cycles and routing said best routable optimized placement using a routing procedure.

10. A method as described in claim 9 wherein said step (d) comprises the step of adjusting placements of each input/output port along its eligible edge placement domain to a placement position closest to a coupled cell that is nearest to said eligible edge placement domain of said each input/output port.

11. A method as described in claim 10 wherein said step of adjusting placements of each input/output port along its eligible edge placement domain comprises the steps of:

identifying a set of cells within a net coupled to said input/output port;

determining which cell of said set of cells is closest to said eligible edge placement domain of said input/output port, said cell being said coupled cell; and

adjusting a placement of said input/output port to said placement position closest to said coupled cell.

12. A method as described in claim 9 wherein said placer procedure of step (c) is a quadratic based placement procedure minimizing wire connections between coupled cells of each circuit block and between input/output ports coupled to cells.

13. A method as described in claim 9 wherein said input/output placement constraints also include input/output port ordering constraints and input/output port maximum density constraints.

14. A method as described in claim 9 wherein said cells of said integrated circuit design are standard cells of variable width and uniform height.

15. A method as described in claim 9 wherein said input/output placement constraints also include an arbitrary initial placement of said input/output ports of said integrated circuit design, said arbitrary initial placement provided by a global floorplanner procedure.

16. A computer system having a processor coupled to a bus, a memory coupled to said bus, and logic, stored in said memory, for causing said computer system to place cells within an integrated circuit design, said logic comprising:

first program instructions receiving said integrated circuit design having circuit blocks which include cells and input/output ports;

second program instructions receiving input/output port constraints including a set of eligible edge placement domains individually associated with said input/output ports;

third program instructions obtaining an arbitrary initial placement for each input/output port within its associated eligible edge placement domain;

fourth program instructions placing said cells of each circuit block using a placer procedure that does not optimize placement of said input/output ports;

fifth program instructions optimizing placements of said input/output ports in accordance with an optimization rule by adjusting placements of said input/output ports within their eligible edge placement domains;

sixth program instructions performing a routability estimate of said integrated circuit design; and

seventh program instructions for repeating said third, fourth and fifth program instructions for n cycles; and

eight program instructions selecting a best routable placement design of said n cycles and routing said best routable placement design.

17. A computer system as described in claim 16 wherein said optimization rule optimizes input/output port placement to reduce wire length and to reduce routing congestion within each circuit block.

18. A computer system as described in claim 17 wherein said fifth program instructions comprises instructions for adjusting placements of each input/output port along its eligible edge placement domains to a placement position closest to a coupled cell that is nearest to said eligible edge placement domain of said each input/output port in order to reduce wire length.

19. A computer system as described in claim 17 wherein said fourth program instructions comprise a quadratic based placement procedure to reduce wire length within each circuit block.

20. A computer system as described in claim 16 further comprising global floorplanner program instructions and wherein said input/output port placement constraints are generated by said global floorplanner program instructions.

21. A computer system as described in claim 20 wherein said input/output port constraints further comprise an arbitrary initial input/output port placement, input/output port ordering constraints, and input/output maximum density constraints.

* * * * *

ATTACHMENT F

Hopper Hierarchical Flow Improves Turnaround in Physical Design of Large IC

by Paul Rodman

Today's problems in chip design are related to flow, not tools. Building an in-house flow—the successful interplay of tools, data and people—has become increasingly difficult because there aren't enough skilled people even as physical designs (such as SoC) keeping growing more complex. And if that isn't enough, there are deep-submicron semiconductor processes to consider, as well as the profusion of tools that have become mandatory. It's clear that engineers need more than just a bag of tools; they need a starting point built upon the collective flow and software expertise from the best-in-class design community.

Take the flow we used to design a high-performance graphics chip for 3dfx Interactive. It's based on proprietary physical-design automation software called Hopper. Hopper makes practical a new automated physical-design flow that offers:

- Abutted hierarchical design,
- Concurrent design,
- Automation of all tasks and easier "what if?" experimentation.

Hopper is essentially an automation engine that enables ReShape to capture physical design flow know-how, with tool-specific knowledge (the best default settings) and design-specific knowledge (tool parameters and event ordering for a specific chip).

Hopper is built on top of commercially available tools such as Avanti's Apollo, Hercules and StarRC-XT, as well as proprietary tools that perform such tasks as repeater insertion and clock distribution. Hopper is essentially an automation engine that enables ReShape to capture physical design flow know-how, with tool-specific knowledge (the best default settings) and design-specific knowledge (tool parameters and event ordering for a specific chip; see Fig. 1, page 28).

Our challenge was to make Hopper meet the following characteristics of the 3dfx graphics chip:

- fabbed using Taiwan Semiconductor Manufacturing Co.'s 0.18-micron process
- six-layer metal
- 1.5 million placeable objects
- 30 million transistors
- 200 RAMs, four PLLs, three D/A converters, two AGPs
- 18 blocks (12 core, four pad-ring blocks)
- 18 different clocks, up to 533 MHz (typically 200 to 350 MHz)
- largest block, 250,000 placeable objects
- over 10,000 repeaters added.

Designs like these are growing so quickly that they are outpacing EDA tool capacity, which makes hierarchical physical design a necessity. The smaller netlists that result from a hierarchical approach translate to shorter run-times, higher tool reliability (fewer core dumps), improved quality of results and greater determinism from run to run.

But more important, in a hierarchical approach the design team gains the benefits of block-level parallelism, which makes it possible to employ people and tools more effectively. Block-level design enables different designers to work in parallel more effectively on the same chip.

A hierarchical flow also supports more deterministic results from run to run. By definition, dividing the chip into blocks limits the potential dispersion of cells and therefore reduces the potential for radical timing changes or congestion changes. However, in a traditional flat flow there is no guarantee that those cells will be located within the same appropriate proximity; therefore, every run with minor changes may cause new problems to surface.

One disadvantage of hierarchical design is that many optimizations don't occur, because the blocks are separated and the changes that must be made are less apparent to the engineer. This is the "horizon effect" and it can result in poor-quality results. A number of tasks suffer from the horizon effect; they are:

- pin assignment
- circuit rules (e.g., max transition)
- timing problems
- verification problems (e.g., antenna rules)
- clock distribution
- power distribution.

ReShape's flow performs hierarchical physical design without these problems. Because our flow uses previous outputs as the input of the next run, along with the latest changes, we are able to see how the blocks fitted together last time and leverage that history to refine our layout. The traditional flow tries to produce the optimal layout from one run; ours allows a series of runs that produce more optimized layouts each time. In a sense, the tools become smarter with each run and are able to use the previous layouts to avoid the horizon effect.

A traditional hierarchical flow relies on channels—open lanes of empty space left between all the blocks—to provide a pathway for connections for last-minute design fixes. Channels are undesirable for three reasons:

- They create potential coupling problems because they implicitly bundle wires. This increases the risk the chip will not run at speed, and it may not run at all.
- Top-level nets travel longer distances since they must go around, rather than through, the blocks to reach their destination. These longer distances can negatively affect timing.
- They waste chip area.

ReShape's flow solves the channel problem by removing channels and using abutted blocks (Fig. 2, page 30). This optimizes block interconnections because signals cut through the blocks, utilizing the extra metal resources within those blocks. Consequently, there are no spaces between blocks. This allows for a more compact physical design and shorter wires, resulting in shorter paths, greater reliability and faster operation.

Hierarchical design enables concurrent physical and logic design. In a traditional flow, the back-end design team must wait until the front-end RTL design team has finished, resulting in both schedule and quality problems. For example, several logical blocks of a chip may already be complete while completion of others may be months away. In a typical flat methodology, the physical-design team would not have access to a complete netlist and therefore experimentation—with meaningful results—would not be possible.

Concurrent design

Without back-end feedback, the functional-design team can unknowingly create problems that are too difficult or costly to solve once the chip goes to physical design. But concurrent functional and physical design permits the physical-design team to start its process as soon as the main structure of some of the netlists is determined, which can be several months (or even a year) before the completion of the entire front-end design.

Starting physical design early enables the front-end team to refine the RTL to address problems generated by the physical-design process. Front-end designers make decisions that affect the physical design; therefore, the ideal methodology would provide them with information about the physical design on which to base those decisions. Early feedback about designs can produce a chip of much higher quality. In fact, with deep-submicron designs needing several repeater insertion delays just to cross the chip, early experimentation with the floor plan becomes essential.

ReShape's flow solves the channel problem by removing channels and using abutted blocks.

Concurrent design makes sense because the main structures of a block-level netlist generally emerge early in the design operation. The remainder of functional-design time is usually spent implementing the control logic, verifying the design and making minor bug fixes, but these changes do not usually have a major impact on the behavior of the netlist in the back end. With this in mind, why not give the physical-design team access to the parts of the logic that are complete and derive the benefits from concurrent design (Fig.3, page 32)?

In our case, concurrent front-end and back-end design enabled the RTL design team to make changes at a more convenient place in the design process and solve back-end problems more efficiently. We found this to be an important advantage.

For example, in the largest block of the design, we were plagued by congestion or hot spots that prevented a clean route. Inspection revealed that the portion of the netlist hierarchy in that area contained an enormous number of high-fanout nets acting as selects to AOI

Hopper components

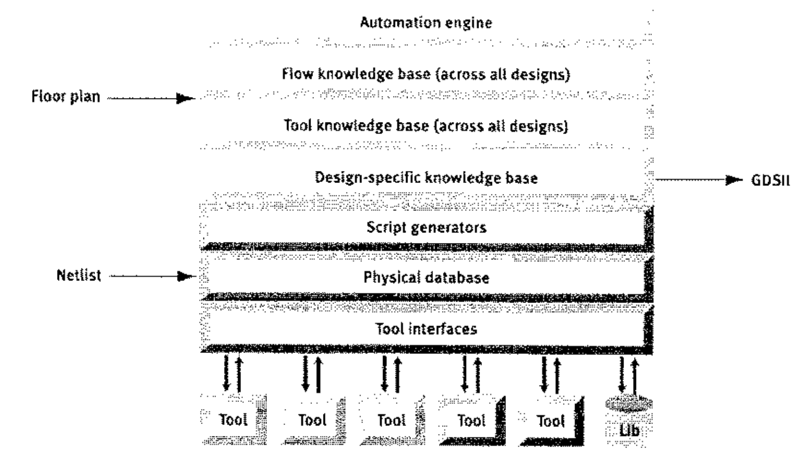


FIG.1 Hopper enables ReShape to capture physical design flow know-how with tool-specific and design-specific knowledge and automatically generates tool scripts specific to the chip.

gates actually functioning as 2:1 multiplexers. Our Synopsys tools had chosen the AOI gate because it looked slightly better on paper.

Two fixes were implemented to solve this problem. We changed the synthesis script to use the infer_mux directive, which reduced the number of high-fanout nets by a factor of two. And we added a pass of buffer tree optimization to the flow for this block

By discovering these kinds of back-end obstacles early, we caught the RTL design team while it could still make changes relatively painlessly.

In deep-submicron designs, the difference between wire model and reality is huge, making them difficult or impossible to use for timing convergence. Some design teams will be conservative and build in margin to cover the difference. Unfortunately, with today's processes this approach often isn't feasible.

The ReShape flow creates per-block wire-load models, which are used for synthesis only. The logic engineer ignores the synthesis timing reports (except for simple A vs. B netlist comparisons), instead converging on the post-placement timing we provide from the flow. (This placement-based timing has been correlated to at least one previous full-route/full-extraction run.)

The wire model is used only to create a netlist with the appropriate level of stiffness for optimal back-end timing convergence. We found that once it is set properly engineers could inject new netlists into the flow and evaluate RTL or synthesis changes with real data in a few hours.

After a run, engineers want to know if the chip is converging on timing and if it has any routing-congestion problems. With the ReShape flow, we did the whole loop in just a few hours for most blocks. For example, a block with 100,000 placeable instances took about 10 hours to completely converge on timing and routing. This same process could take days in a traditional methodology. At some point the RTL converges on a final set of netlists and the push to tapeout is on.

With the ReShape flow, we did the whole loop in a few hours for most blocks, a job that could take days otherwise.

Due to the automation and the hierarchical design process, we were able to build the entire 3dfx chip from scratch in 24 hours. Starting from gate-level netlists (netlists alone were over 1 Gbit) and with the previous floor plan and flow configuration checked out from the source tree, we spawned more than 4,000 individual jobs, with all blocks placed and routed and timing converged. The ReShape tools and the Avanti runs created more than 10,000 files.

If network or hardware problems caused a crash, the flow could be automatically restarted and would resume execution where it left off.

Automatic steps

One of the most important advantages of the ReShape flow is the automation of thousands of manual steps normally required for a hierarchical physical design. The flow provides a framework for adding special-value automation in incremental stages to solve the many problems that come up while building a block. We have identified time-consuming tasks that could be automated and we have developed code that's incorporated into the flow to handle those tasks. Automation not only saves a significant amount of design time, but it is also based on previous chip successes and on proven configurations, enabling a design team to fine-tune these settings over time to ensure the highest tool performance.

Inherited knowledge

For example, we divide the placement process into several discrete steps. The first is preparing the command file. Our flow opens our database and studies the block, using controls that have been defined according to user preferences, then automates the production of a command file that encapsulates all our learning about the best way to perform the placement tasks for that block. Any person who uses the flow inherits the benefits of any knowledge from the flow builders—and perhaps that block's previous builder.

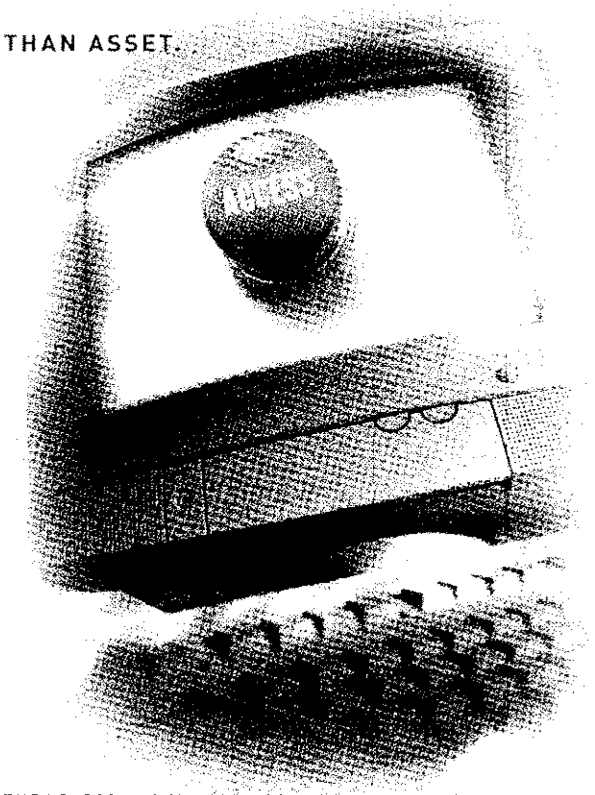
Another example is the automation of log file review. The log file is a critical line of communication from vendor tools to inform the user about the results of each task execution. If you do not go through every line in the log file, you may miss a single-line message, among tens of thousands of lines, that indicates a problem. The sad implications of this message may not be obvious for days or weeks. The ReShape flow has embedded log-checking software that automatically opens and reads the log, looking for indications of errors and highlighting them, and stopping the process.

No matter how good design tools are, new physical-design challenges usually emerge on new projects or with a new library or process. By nature, EDA vendors can address the needs of only some of them. But the ReShape flow creates special-purpose code to deal with special needs, then configures and "clicks it in" the flow. The flow acts as powerful framework for adding such tools.

NOBODY GIVES YOU

ACCESS TO MORE

THAN ASSET.



THE LEADER IN BOUNDARY-SCAN TECHNOLOGY. Today's advanced devices have created a variety of limited access challenges. At ASSET, our solutions address this by giving you virtual access to the board. But we don't stop there. We also give you complete access to on-line technical support, training and the expertise of seasoned professionals that you need to reap the full benefits of boundary scan. It's all a part of our commitment to accelerate your time to market and ensure a quicker return on your investment.

Visit us at www.asset-intertech.com/access and get access to more.

ASSET®
THE BOUNDARY-SCAN COMPANY

TAB C

AO 88 (Rev. 11/94) Subpoena in a Civil Case

Issued by the

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SYNOPSYS, INC., a Delaware corporation,

v.

MAGMA DESIGN AUTOMATION, a Delaware corporation

SUBPOENA IN A CIVIL CASE

Case Number:[1] 05-701 GMS (DISTRICT OF DELAWARE)

TO: Pillsbury Winthrop Shaw Pittman LLP, 2475 Hanover Street, Palo Alto, CA 94304

[ ] YOU ARE COMMANDED to appear in the United States District Court at the place, date, and time specified below to testify in the above case.

PLACE OF TESTIMONY	COURTROOM
	DATE AND TIME

[ ] YOU ARE COMMANDED to appear at the place, date, and time specified below to testify at the taking of a deposition in the above case.

PLACE OF DEPOSITION	DATE AND TIME

[X] YOU ARE COMMANDED to produce and permit inspection and copying of the following documents or objects at the place, date, and time specified below (list documents or objects):

Please see Attachment A.

PLACE	DATE AND TIME
Dechert LLP, 1117 California Street, Palo Alto, CA 94304; (650) 813-4800	November 17, 2006 5:00 p.m.

[ ] YOU ARE COMMANDED to permit inspection of the following premises at the date and time specified below.

PREMISES	DATE AND TIME

Any organization not a party to this suit that is subpoenaed for the taking of a deposition shall designate one or more officers, directors, or managing agents, or other persons who consent to testify on its behalf, and may set forth, for each person designated, the matters on which the person will testify. Federal Rules of Civil Procedure, 30(b)(6).

ISSUING OFFICER'S SIGNATURE AND TITLE (INDICATE IF ATTORNEY FOR PLAINTIFF OR DEFENDANT)	DATE
[illegible] Attorneys for Plaintiff SYNOPSYS, INC.	October 25, 2006

ISSUING OFFICER'S NAME ADDRESS AND TELEPHONE NUMBER

Daniel B. Epstein, Dechert LLP, 1117 California Avenue, Palo Alto, CA 94304
Telephone: 650-813-4800

(See Rule 45, Federal Rules of Civil Procedure, parts C & D on reverse)

[1] If action is pending in district other than district of issuance, state district under case number.

AO 88 (Rev. 11/94) Subpoena in a Civil Case

PROOF OF SERVICE

	DATE	PLACE
SERVED		

SERVED ON (PRINT NAME) MANNER OF SERVICE

SERVED BY (PRINT NAME) TITLE

DECLARATION OF SERVER

I declare under penalty of perjury under the laws of the United States of America that the foregoing information contained in the Proof of Service is true and correct.

Executed on ______________________________
DATE

SIGNATURE OF SERVER

ADDRESS OF SERVER

Rule 45, Federal Rules of Civil Procedure, Parts C & D:

(c) PROTECTION OF PERSONS SUBJECT TO SUBPOENAS.

(1) A party or an attorney responsible for the issuance and service of a subpoena shall take reasonable steps to avoid imposing undue burden or expense on a person subject to that subpoena. The court on behalf of which the subpoena was issued shall enforce this duty and impose upon the party or attorney in breach of this duty an appropriate sanction which may include, but is not limited to, lost earnings and reasonable attorney's fee.

(2) (A) A person commanded to produce and permit inspection and copying of designated books, papers, documents or tangible things, or inspection of premises need not appear in person at the place of production or inspection unless commanded to appear for deposition, hearing or trial.

(B) Subject to paragraph (d) (2) of this rule, a person commanded to produce and permit inspection and copying may, within 14 days after service of subpoena or before the time specified for compliance if such time is less than 14 days after service, serve upon the party or attorney designated in the subpoena written objection to inspection or copying of any or all of the designated materials or of the premises. If objection is made, the party serving the subpoena shall not be entitled to inspect and copy materials or inspect the premises except pursuant to an order of the court by which the subpoena was issued. If objection has been made, the party serving the subpoena may, upon notice to the person commanded to produce, move at any time for an order to compel the production. Such an order to comply production shall protect any person who is not a party or an officer of a party from significant expense resulting from the inspection and copying commanded.

(3) (A) On timely motion, the court by which a subpoena was issued shall quash or modify the subpoena if it

(i) fails to allow reasonable time for compliance,

(ii) requires a person who is not a party or an officer of a party to travel to a place more than 100 miles from the place where that person resides, is employed or regularly transacts business in person, except that, subject to the provisions of clause (c) (3) (B) (iii) of this rule, such a person may in order to attend trial be commanded to travel from any such place within the state in which the trial is held, or the demanding party to contest the claim.

(iii) requires disclosure of privileged or other protected matter and no exception or waiver applies, or

(iv) subjects a person to undue burden.

(B) If a subpoena

(i) requires disclosure of a trade secret or other confidential research, development, or commercial information, or

(ii) requires disclosure of an unretained expert's opinion or information not describing specific events or occurrences in dispute and resulting from the expert's study made not at the request of any party, or

(iii) requires a person who is not a party or an officer of a party to incur substantial expense to travel more than 100 miles to attend trial, the court may, to protect a person subject to or affected by the subpoena, quash or modify the subpoena, or if the party in who behalf the subpoena is issued shows a substantial need for the testimony or material that cannot be otherwise met without undue hardship and assures that the person to whom the subpoena is addressed will be reasonably compensated, the court may order appearance or production only upon specified conditions.

(d) DUTIES IN RESPONDING TO SUBPOENA.

(1) A person responding to a subpoena to produce documents shall produce them as they are kept in the usual course of business or shall organize and label them to correspond with the categories in the demand.

(2) When information subject to a subpoena is withheld on a claim that is privileged or subject to protection as trial preparation materials, the claim shall be made expressly and shall be supported by a description of the nature of the documents, communications, or things not produced that is sufficient to enable the demanding party to contest the claim.

ATTACHMENT A

Pursuant to Rules 34 and 45 of the Federal Rules of Civil Procedure, Plaintiff SYNOPSYS, INC. ("SYNOPSYS") requests that PILLSBURY WINTHROP SHAW PITTMAN LLP ("PILLSBURY") produce and permit the inspection and copying of the documents or tangible things described below in its possession, custody, or control.

INSTRUCTIONS

1. Unless otherwise noted, this set of demands requires the production of documents or tangible things that were prepared, created, written, sent, dated or received at any time up to the present.

2. In producing documents or tangible things pursuant to these demands, PILLSBURY must conform to the requirements of Federal Rule of Civil Procedure 34(b). This means that PILLSBURY shall produce documents as they are kept in the usual course of business or shall organize and label the documents to correspond with the categories in the document requests.

3. If PILLSBURY withholds any documents or tangible things under a claim of privilege, please furnish with the response to these demands a privilege and/or redaction log identifying each document or tangible thing for which privilege is claimed, including the following information:

a. The date, sender, recipient, and subject matter of the documents or tangible thing;

b. The basis upon which privilege is claimed;

c. The paragraphs, paragraph, or subparts of the demand to which the document or tangible thing corresponds.

DEFINITIONS

1. "PILLSBURY" means PILLSBURY WINTHROP SHAW PITTMAN LLP and any of its predecessors, successors, assigns, agents, employees, officers, directors, affiliates, partners, subsidiaries, parent-corporations, investors, attorneys, or other persons or entities acting on its behalf.

2. "PTO" means the United States Patent and Trademark Office.

3. "DOCUMENT" means any "writing," as defined by Federal Rule of Evidence 1001, in PILLSBURY'S possession, custody or control (including the original and all non-identical copies thereof) regardless of where located and including, but not limited to, files, file folders, contracts, agreements, financial statements, ledger sheets or other accounting records, loan papers, inventory records, tape recordings, transcripts, drawings, correspondence, communications, reports, studies, summaries, indices, memoranda, calendar or diary entries, handwritten notes, working papers, minutes, agenda, bulletins, notices, announcements, instructions, charts, manuals, brochures, schedules, telegrams, teletypes, films, videotapes, photographs, microfilm or microfiche, letters, faxes, computer records, computer disks, magnetic tape, optical disks, electronic backups, source code, object code, executable files, library definitions, documentation, tutorial materials, user's guides, reference materials, installation notes, implementation details, man pages, and any other documents or data compilations.

4. "COMMUNICATION" means any transfer of information, ideas, opinions or thoughts by any means, written, oral or otherwise, at any time or place under any circumstances. The definition is not limited to transfers between persons but also includes other transfers, such as records and memoranda to file; any written letter, memorandum, or other document

which was sent by one or more individuals to another or others; any telephone call between one or more individual and another or others, whether such call was by chance or prearranged, formal or informal; any electronic mail ("email") messages sent by one or more individuals to another or others; and any conversation or meeting between one or more individuals and another or others, whether such contact was by chance or prearranged, formal or informal. Unless otherwise indicated, a request for COMMUNICATIONS includes PILLSBURY'S internal COMMUNICATIONS and COMMUNICATIONS between PILLSBURY and any other person.

5. "'745 PATENT" means United States Patent No. 6,519,745, Application No. 09/579,966, and any continuation applications, continuation-in-part applications, and divisional applications filed thereon, or foreign counterparts.

6. "PRIOR ART" is used as that term is generally understood in proceedings before the PTO. PRIOR ART includes all information that was subject to a duty of disclosure to the PTO with respect to the application leading to the '745 PATENT, and all information now known to PILLSBURY that would have been subject to a duty of disclosure to the PTO had that information been known to the named inventors or to the prosecuting attorney during the prosecution of the '745 PATENT. PRIOR ART also refers to, by way of example and without limitation, to the subject matter described in 35 U.S.C. §§ 102 and 103, including, but not limited to, (1) publications physical devices, prototypes, uses, sales, and offers for sale and any DOCUMENT or thing evidencing any of the foregoing and/or (2) any DOCUMENT or thing that any person has at any time suggested, or contended invalidates (or may invalidate), anticipates (or may anticipate), or makes obvious (or may make obvious), alone or in combination with other DOCUMENTS or things (or the skill or knowledge of a person of

ordinary skill in the art), (a) any claim of the '745 PATENT, or (b) any claim pending during the prosecution of the '745 PATENT.

7. "RELATING TO" means constituting, containing, consisting of, composing, comprising, embodying, referring to, summarizing, discussing, showing, commenting upon, or describing.

<u>**REQUESTS FOR PRODUCTION**</u>

<u>**REQUEST FOR PRODUCTION NO. 1:**</u>

All DOCUMENTS RELATING TO prosecution in the PTO of the '745 PATENT.

<u>**REQUEST FOR PRODUCTION NO. 2:**</u>

All COMMUNICATIONS RELATING TO the '745 PATENT.

<u>**REQUEST FOR PRODUCTION NO. 3:**</u>

All DOCUMENTS RELATING TO any sale, offer for sale, public use or disclosure of any system, hardware, software, product, or method covered by any claim of the '745 PATENT.

<u>**REQUEST FOR PRODUCTION NO. 4:**</u>

All DOCUMENTS RELATING TO PRIOR ART to the invention of any claim of the '745 PATENT or to any invention of any claim pending during the prosecution of the '745 PATENT, cited to, or by, the named inventors of the '745 PATENT.

<u>**REQUEST FOR PRODUCTION NO. 5:**</u>

All DOCUMENTS RELATING TO all PRIOR ART searches conducted in connection with the prosecution of the '745 PATENT.

<u>**REQUEST FOR PRODUCTION NO. 6:**</u>

All DOCUMENTS RELATING TO the conception and/or reduction to practice of the '745 PATENT.

REQUEST FOR PRODUCTION NO. 7:

All DOCUMENTS RELATING TO any invention disclosures from any of the named inventors of the '745 PATENT.

REQUEST FOR PRODUCTION NO. 8:

All DOCUMENTS RELATING TO COMMUNICATIONS between PILLSBURY and the named inventors of the '745 PATENT.

REQUEST FOR PRODUCTION NO. 9:

All DOCUMENTS RELATING TO any advantage or improvement provided by the inventions claimed in the '745 PATENT.

REQUEST FOR PRODUCTION NO. 10:

All DOCUMENTS RELATING TO the existence of any problem with which the inventions claimed in the '745 PATENT are concerned, any proposed or actual solution thereto.

REQUEST FOR PRODUCTION NO. 11:

All DOCUMENTS RELATING TO any long-felt need for the inventions claimed in the '745 PATENT and the satisfaction thereof, including all DOCUMENTS RELATING TO any commercial success of the inventions claimed in the '745 PATENT.

REQUEST FOR PRODUCTION NO. 12:

All DOCUMENTS RELATING TO the date when each of the inventions claimed in the '745 PATENT began to be developed, including documents sufficient to determine such date.

REQUEST FOR PRODUCTION NO. 13:

All DOCUMENTS and tangible things RELATING TO the research, development, or design of any of the inventions claimed in the '745 PATENT.

REQUEST FOR PRODUCTION NO. 14:

All DOCUMENTS and tangible things RELATING TO the modification, improvement, upgrade, construction, manufacture, assembly or testing of any of the inventions claimed in the '745 PATENT.

REQUEST FOR PRODUCTION NO. 15:

All DOCUMENTS and tangible things RELATING TO the ownership of or interest in the '745 PATENT, including all assignments, notices, notices or records RELATING TO the assignments, in the PTO or otherwise.

REQUEST FOR PRODUCTION NO. 16:

All DOCUMENTS RELATING TO telephone or in-person interviews with the PTO examiner for the '745 PATENT.

REQUEST FOR PRODUCTION NO. 17:

All DOCUMENTS RELATING TO the prosecution history of the '745 PATENT.

REQUEST FOR PRODUCTION NO. 18:

All DOCUMENTS RELATING TO the preparation of the application which matured into the '745 PATENT.

REQUEST FOR PRODUCTION NO. 19:

All DOCUMENTS RELATING TO any analysis, opinion, construction, or interpretation of any of the claims – or terms within the claims – of the '745 PATENT.

CERTIFICATE OF SERVICE

The undersigned hereby certifies that on October 30, 2006 she caused to be electronically filed the foregoing with the Clerk of the Court using CM/ECF, which will send notification of such filing(s) to:

William J. Marsden, Jr.
Fish & Richardson, P.C.

I also certify that copies were caused to be served on October 30, 2006 upon the following in the manner indicated:

BY HAND & E-MAIL

William J. Marsden
Fish & Richardson, P.C.
919 N. Market Street
Suite 1100
Wilmington, DE 19801

BY E-MAIL

James Pooley
Pooley & Oliver LLP
Five Palo Alto Square
3000 El Camino Real
Palo Alto, CA 94306-2109

/s/ Leslie A. Polizoti
Leslie A. Polizoti (#4299)
MORRIS, NICHOLS, ARSHT & TUNNELL LLP
Wilmington, DE 19801
(302) 658-9200
lpolizoti@mnat.com